The University of Jordan

Authorization Form

I, Nancy Jamil Khrais , authorize the University of Jordan to supply copies of my Thesis/ Dissertation to libraries or establishments or individuals on request, according to the University of Jordan regulations.

Signature:

Date:

الجامعة الأردنيــــة
كلية الدراسات العليا

التاريـــخ: / /

نموذج رقم (١٨)
اقرار والتزام بالمعايير الأخلاقية والأمانة العلمية
وقوانين الجامعة الأردنية وأنظمتها وتعليماتها
لطلبة الماجستير

أنا الطالب: نانسي جميل أحمد خريس الرقم الجامعي: (٨٠٨٠٩٨٥)

تخصص: أنظمة المعلومات الحاسوبية الكليـــــة: تكنولوجيا المعلومات

عنوان الرسالة:

Database Interoperability in collaborative Networks

اعلن بأنني قد التزمت بقوانين الجامعة الأردنية وأنظمتها وتعليماتها وقراراتها السارية المفعول المتعلقة باعداد رسائل الماجستير عندما قمت شخصيا" باعداد رسالتي وذلك بما ينسجم مع الأمانة العلمية وكافة المعايير الأخلاقية المتعارف عليها في كتابة الرسائل العلمية. كما أنني أعلن بأن رسالتي هذه غير منقولة أو مستلة من رسائل أو كتب أو أبحاث أو أي منشورات علمية تم نشرها أو تخزينها في أي وسيلة اعلامية، وتأسيسا" على ما تقدم فانني أتحمل المسؤولية بأنواعها كافة فيما لو تبين غير ذلك بما فيه حق مجلس العمداء في الجامعة الأردنية بالغاء قرار منحي الدرجة العلمية التي حصلت عليها وسحب شهادة التخرج مني بعد صدورها دون أن يكون لي أي حق في التظلم أو الاعتراض أو الطعن بأي صورة كانت في القرار الصادر عن مجلس العمداء بهذا الصدد.

توقيع الطالب:............................ التاريخ: ٢ / ٥ / ٢٠١١

تعتمد كلية الدراسات العليا
هذه النسخة من الرسالــة
التوقيع...... التاريخ ٢٠١١/٥/٢

DATABASE INTEROPERABILITY IN COLLABORATIVE NETWORKS

By
Nancy Jamil khrais

Supervisor
Dr. Nadim Obeid, Prof.

This Thesis was submitted in Partial Fulfillment of the Requirements for the Master's Degree in computer information systems

Faculty of Graduate Studies
The University of Jordan

May, 2011

COMMITTEE DECISION

This Thesis /Dissertation (**Database Interoperability in Collaborative Networks**) was successfully Defended and Approved on 11/5/2011.

Examination Committee	Signature
Dr. Nadim Obeid (Supervisor) Professor of Artificial Intelligence	
Dr. Mohammad Belal Al-Zoubi (Member) Professor of Data Mining / GIS and Computer Graphics	
Dr. Thair Hamtini (Member) Assistant Professor of E-Learning	
Dr. Mohammad Al-Abbadi (Member) Associate Professor of Multimedia and Image Processing. (Mutah University)	

ACKNOWLEDGEMENTS

I would like to sincerely thank almighty Allah for all his grants that he bestowed on me.

My special acknowledgements go to my supervisor professor Dr. Nadim Obeid for giving me timely advice, encouragement and guidance through out the project. I am grateful for his patience in listening to my ideas and giving his valuable inputs that enabled me to develop an understanding of the subject. I am grateful for his words of encouragement that kept my spirits high.

I would like to present my sincere thankfulness to my dear father and mother for their great role in my life and their numerous sacrifices for me and for my brothers and sisters. Many thanks for my family for their moral support because without it, I would not have been able to accomplish anything.

Lastly, I offer my regards and blessings to all of those who supported me in many situations during my study. A special thanks to my friends and my coworkers in the ministry of justice especially those in the software development department for their support.

DATABASE INTEROPERABILITY IN COLLABORATIVE NETWORKS

By
Nancy Jamil khrais

Supervisor
Dr. Nadim Obeid, Prof.

ABSTRACT

Information systems for large organizations today are most frequently implemented on a distributed architecture. These systems are independently designed, developed and administered. Different types of heterogeneity as syntactic heterogeneity, semantic heterogeneity and structural heterogeneity make the data integration process more difficult. In light of this situation, we propose an integrative approach to database interoperability that effectively tackles the problems of heterogeneity by using ontology notion and schema template. Our approach is based on four processes: mapping, unification, data definition language (DDL) auditing and schema exchange using schema template. Mapping is a semi-automated process that uses ontologies to link a database schema with a conceptual framework—named virtual schema. Unification is an automated process for integrating ontologies and hence the database to which they are linked. DDL auditing is an automatic process used to handle the changes that may occur on the physical data source structure and inform the system administrator that some problem should be solved. Schema template is used to represent a class of different database schemas sharing the same structure. The goal of using the schema template is the translation of any database whose schema conforms to the source template into a format conforming to the target template. Using these methods, we employed our approach to interoperate two different data sources related to legal domain which are Mizan database and police database to know the criminal curriculum vitae (C.C.V) for each citizen in the country. In addition, we integrate a civil status system with our system to get the most correct personal data about the citizen.

LIST OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

DBMS	Database Management System
NCRC	Non-Criminal Record Certificate
C.C.V	Criminal Curriculum Vitae
NLP	Natural Language Processing
GAV	Global As View
LAV	Local as View
GUI	Graphical User Interface
DDL	Data Definition Language
KM	Knowledge Management

1 Introduction

1.1 Background

In the last years, we have witnessed an increasing complexity of database applications, especially when several, possibly autonomous data sources need to be accessed, transformed and combined. Integrating a number of information systems is difficult because each system has two main characteristics:

1) Autonomy: Each database was designed independently. In other words, the database designer did only consider the problem/domain with which it is concerned without any consideration to share data or execute queries with any other system. Hence, it is not at all obvious that the database can interoperate (e.g., share data, execute queries jointly) with other databases without too much effort.

2) Heterogeneity: As shown in figure 1, there are Differences type of heterogeneity , differences in hardware, operating systems, database management system DBMS (systems heterogeneity), differences in data model and data languages (syntactic heterogeneity), and differences in how the real world is perceived, conceptualized and represented in the databases (semantic heterogeneity). These heterogeneity types make the data integration more difficult and cause duplication of efforts and resources in order to collect the required data.

New technologies are being created to facilitate data search, access, retrieval and gathering from heterogeneous data sources. Ontology is one of these technologies which can used to provide semantic cooperation of heterogeneous sources.

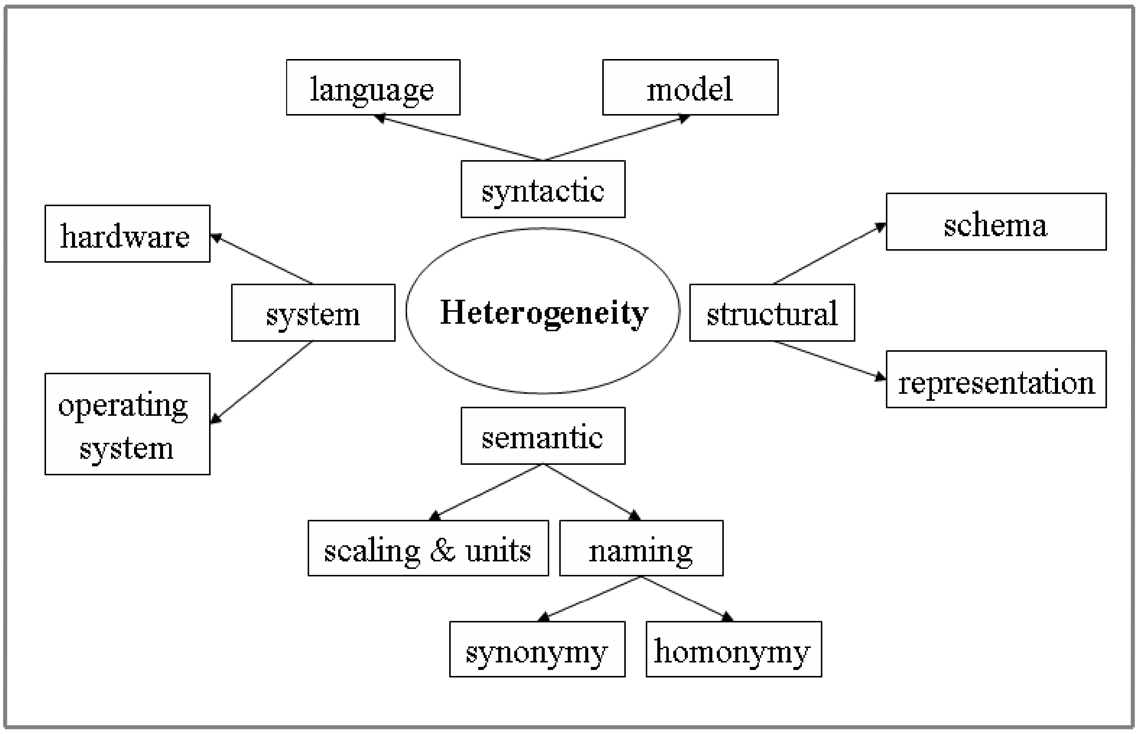


Figure 1 : Heterogeneity Types

In data integration process, two questions need to consider are; how can we detect that the underlying data sources have changed. Second question is how we can integrate multiple databases in a simple way when the number of integrated systems is increased? Detecting changes on a source database structure as adding, deleting columns or dropping tables is very important because these changes can affect the structure of a virtual schema defined with .so we will build DDL auditing on schema level in each data source to capture the changes and reflect these changes to the virtual schema. The goal of auditing is to track the changes on database schema at any time.

Building a template that contains table's names, attributes, keys and relations, in addition to defining specific rules in each data source (as see in chapter 2) can facilitate integrating new source to our interoperability system .

1.2 Problem Definition

Today, there are many examples of scenarios in which data residing at different sources must be accessed in a unified way overcoming the whole types of heterogeneities. Some of examples where integrated view is useful are given below.

• Comparison-shopping applications that compare the prices and specifications of various items. Data about items is accessed from different heterogeneous data sources so we have a need for the integrated view of all the sources and a query mechanism.

• Any application that needs to interact with data from two or more sources .For example, in the ministry of justice there is a need to know the criminal curriculum vitae (C.C.V) for each citizen in the country. Judges need this information in order to check the reputation for witnesses in some cases (lawsuits) or criminals. Also we need it to issue a NCRC (Non-Criminal Record Certificate) for anyone in the country for different purposes such as employment. As shown in figure 2, the criminal curriculum vitae system need to collect data from multiple databases e.g. Mizan application (MIZAN is a Court Case Management System that automates most routine clerical procedures performed by court staff in the Ministry of Justice), Border & residency and police security (see appendix A , criminal CV information systems). Each database owner is responsible for a specific kind of crimes so we need to integrate these databases until we can reach to the required data from there original data sources in a unified way.

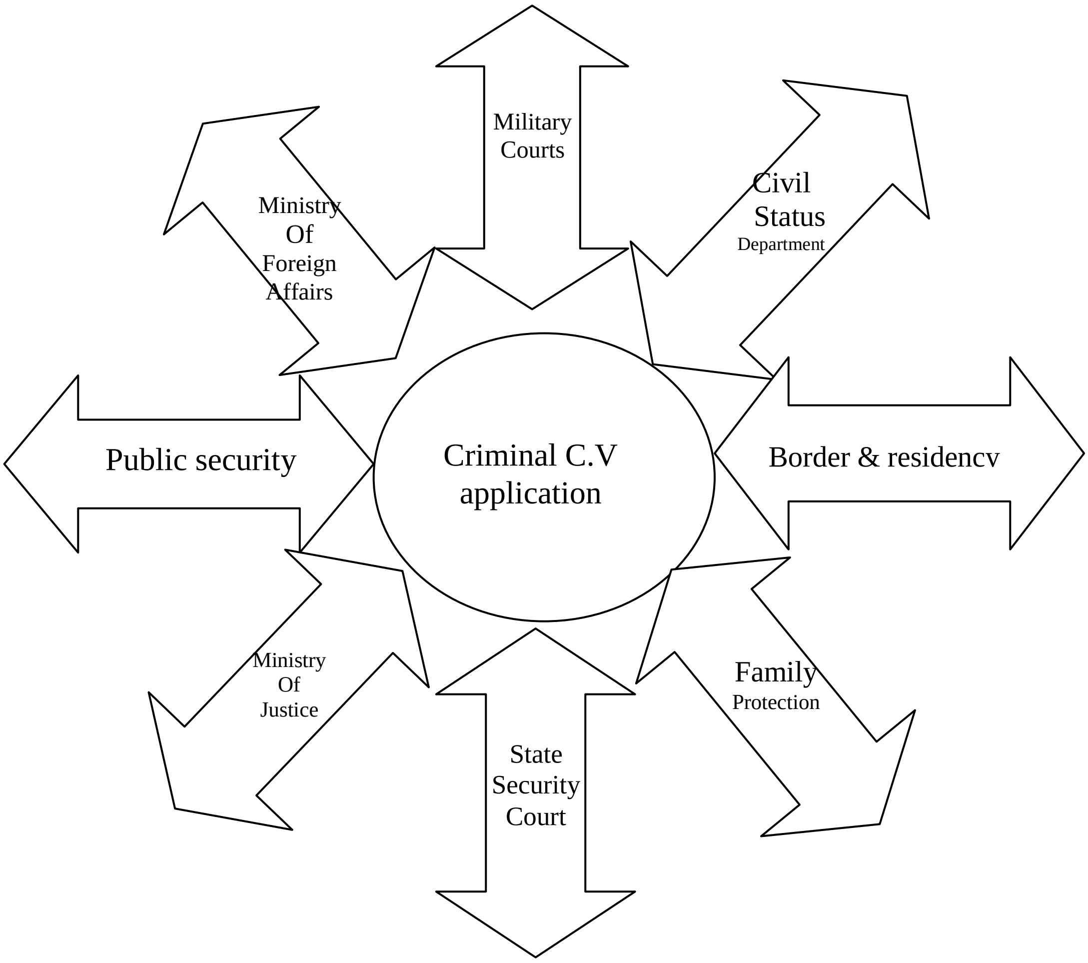


Figure 2 : Multiple Database Sources Related to Criminal C.V Application

The traditional way for doing the integration process needs a lot of manual work during its different stages of integration. First, systems analysts have to know where to look for the required data. They must understand each database schema structure that will be integrated.

In order to know the tables that have the required data and to understand the databases structure, the analysts need to cooperate with databases owners to achieve this task, at the end of this stage the tables that have all the needed data should be determined. The different definitions of databases schemas are frequently conflicting. So, we must solve these conflicts manually by adding new fields, changing a data type or concatenating several tables' fields, etc. After that the data from remote heterogeneous database are

copied on a local server then graphical user interface (GUI) based on the integrated database is designed. This approach is non efficient, complicated and inflexible especially when the number of integrated schemes are increased ,when the owners of the databases schemas need to change their schema structure and when the application area is sensitive to relatively short time delay. The other problems which may arise when using this approach are:

1. The data transferring takes a lot more time and patience.
2. Required data is frequently changed so we need to reload the data from its sources according to the new requirements.
3. Misunderstanding of the source database's structure may cause the need to reload the data.
4. The data is not up to date.
5. There is a need for detecting any change in data sources structure in order to rewrite the scripts that used to transfer the data from it's source.

So, there is a consequent growing need for advanced tools and flexible techniques supporting the management, the exchange, and the integration of different and heterogeneous sources of information.

1.3 Summary of Contribution

The primary objective of this thesis is to provide a mechanism that effectively tackles the database integration problems, allows heterogeneous systems to interact, interoperate and exchange information in a meaningful way, and provide homogeneous access and integrated interface to the heterogeneous and distributed databases. In order to achieve these objectives we will use ontology notion and schema template.

First, two different databases schema (Mizan database & Police database) which have related data are read and analyzed. Then the ontology is created by selecting the best representation for the common concepts, attributes and relations between the two schemas. After identifying the tables and attributes of the databases which are related to each other, the conceptual schema which called virtual schema is built in each side (Mizan database & Police database) using the ontology which is built previously. After that we unified these virtual schemas in a unification schema and using it to build our graphical user interface. We use DDL auditing to handle the changes that may occur on the physical data source structure and inform the system administrator that some problem should be solve.

1.4 Thesis Outline

The rest of this thesis is organized as follows:

In chapter 2, we present different types of conflicts that may be encountered when merging the data from different heterogeneous data sources. In addition, we present the interoperability benefits and the requirements that we need to properly build the interoperability systems. Then we explain the concept of ontology and its main components. Afterwards, the chapter suggests applying ontologies as a solution for database interoperability. In chapter 3, we present two types of approaches being developed to achieve the data integration; syntactic approaches and semantic approaches.The main features, advantages, and disadvantages of these approaches are explained in this chapter. The criteria that can be used to determine the quality of database integration approaches are described in this chapter too.In chapter 4 we show the steps that followed to build interoperable databases (Mizan and police) in details. In chapter 5, we present an evaluation of the achievements. Then we summarize the abilities and the weaknesses of our approach. Finally, we present the possible future improvements along with conclusions.

2 Database Integration

2.1 Introduction

Autonomous organizations define their schemas differently. These different definitions are frequently conflicting and thus their matching and integration is a difficult process. Several new technologies and approaches are being developed to make the data integration process more efficient and flexible such as: ontology, schema template and interoperability. Interoperability is a mechanism that enables the different information systems to communicate and work together in a distributed environment. Many disciplines now develop standardized ontologies that domain experts can use to share and annotate information in their fields. The ontology can be used in interoperability system to solve the problem of semantic heterogeneity. The rest of this chapter is organized as follows: in section 2.2 we describe different type of database integration problems. Section 2.3 presents interoperability mechanism benefits and the requirements that need to properly build interoperability systems. Section 2.3 tells us about ontology and its main components and benefits; also this section presents some methodologies for building the ontology, whereas section 2.4 explains the notion of schema template.

2.2 Database Integration Problems

Autonomous organizations define their schemas differently. These different definitions are frequently conflicting and thus their matching and integration is a challenge as determined below. Different types of conflicts exist among the representation of the same concept in different databases. Each of the following examples represents different kinds of database conflicts that can face the database designers when they are trying to make integration between more than two schemas. In the following example, we tried to integrate Mizan schema police schema (see appendix B). Mizan is an application system used in all Jordanian courts by judges that contains all the cases and their related data such as the claim parties and case status whether it has a final judgment or not. Clearly, the varieties of types of conflicts that exist among databases are not limited to the ones given here.

- Domain Conflicts or Data Representation conflicts: Two attributes may have the same name but have different datatypes. e.g. Databases using different values to represent same concept as "participant_id" field in "Mizan schema" using varchar2(15) data type while the same field in "Police schema " using number (10) data type. So we need to build a new field which can store the two types of integrated fields. To resolve such conflicts, datatypes are expanded through the use of constraint facet redefinition (i.e. adjustment of scale), or through union of disjoint data types only so far as necessary to satisfy boundary conditions. For example "FINE_AMT" field in "Mizan schema" defined as having an integer datatype while the same field in "Police schema" defined as having a decimal datatype. Since decimal and integer are not disjoint sets, this conflict is resolved through the

adjustment of scale. In this case we can assign decimal as the datatype for the global attribute.

- Missing attributes. For example, in "police schema", judgment (year, month, day) is existed but in "Mizan Schema" is missing.
- Missing data. For example, national number in "Police schema” is a required field while it is not required in "Mizan schema", which will cause failure in the process of transforming the data.
- Relationship Conflict: This conflict is related to the case, in which relationships between entities in different schemas are defined differently (Ozgul and Afsarmanesh, 2006). For example, Judjment according to (personal_id) field is founded in one table named "JUD_SENTENCES" in the "police schema" while it is founded in "Mizan schema" by connecting the two tables "CASE_MAIN_PARTY", " JUDGMENT" to get this information.
- Very huge amount of data should be transferred and some time this process is failed because of network problems.
- We don’t have data up to date; there can be time delays in making the data available to system users.
- Lookup table’s problem: different lookup tables between "police schema" and " Mizan schema" make the mapping process between these tables more difficult.
- Key Conflict: This conflict arises when different keys are assigned for the same entity in different schemas (Ozgul and Afsarmanesh, 2006). In the example shown in Figure 3, "JUDGMENT TABLE" In Mizan schema has a primary key consists of one field "CASE ID", while in "police schema" the similar Table

"JUD_SENTENCES" has a primary key consists of two fields (CASE ID, ACSPRS_ID)

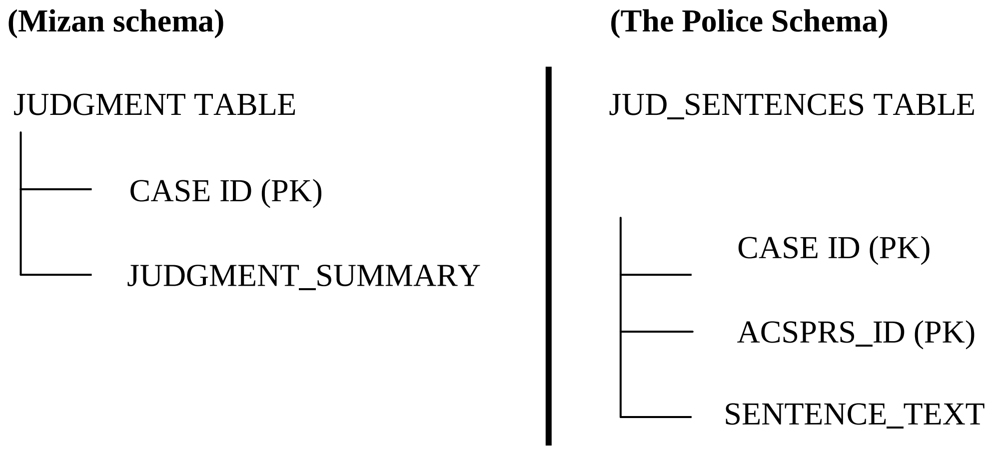


Figure 3 : key Conflict.

- Attribute-Attribute Conflict: This conflict arises when the same concept is represented by using different number of attributes in different schemas (Ozgul and Afsarmanesh ,2006) . The example shown in Figure 4, "judgment_summary" field contains data stored in one column of "judgment" table in "Mizan schema", while in "police schema" the judgment is distributed over five columns (sentence text, year, months, days, fine_amt).

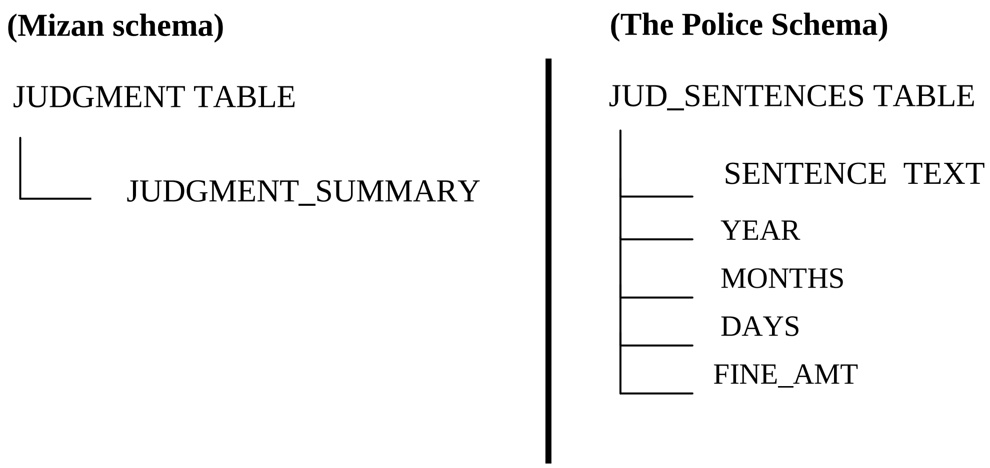


Figure 4 : Attribute-Attribute Conflict.

- Linguistic Conflicts: Linguistic conflicts arise because of different terminology and names that different organizations use to refer to the same data. Linguistic conflicts can be of two types: syntactic and semantic (Ozgul and Afsarmanesh, 2006).

  Also we can refer to this type of conflict as Synonym Naming Conflict and Homonym Naming Conflict. Synonym Naming Conflict means Synonym Schema elements have different names but the same definitions. For example "PARTICIPANT TABLE" In Mizan schema has a attribute "GENDER", while in "police schema" the similar Table "SYS_PERSONS_SJL" has a attribute called "SEX". "PARTICIPANT TABLE" In Mizan schema has a attribute "SECOND_NAME" while in "police schema" the similar Table "SYS_PERSONS_SJL" has a attribute called "FATHER_NAME".

  Homonym Naming Conflict means Homonym Schema elements have the same name but different definitions.

- A data model conflict occurs when the input schemas are defined with different data models (Entity Relationship, Relational, Object-Oriented ...etc).
- Scale conflicts, which involve the use of different units of measure. For example, body temperature can be recorded in degrees of Fahrenheit in one schema and Celsius degrees in the other.

Hence, we need a mechanism that can solve the problems of data integration process , efficiently collect the related data from multiple sources, minimize the time needed to access redundant sources, and respond flexibly when some sources are unavailable.

2.3 Interoperability

Interoperability is the ability of different information systems to communicate and work together in a distributed environment.

There are two important requirements for interoperability. First, systems must understand the data exchanged. This understanding requires shared data models which depend on standard vocabularies, this is called semantic interoperability. Systems must be able to exchange data through shared interface, this is called syntactic interoperability.

2.3.1 Interoperability Benefits

From our experience, we note that there are several major benefits of using interoperability systems, some of these benefits are:

1. Interoperability system can minimize delay when conveying information and can retrieve the information up to date. These factors can be critical in mission areas, such as air defense, where mission effectiveness is sensitive to relatively short time delays and errors in target identification.
2. Minimize the likelihood of errors introduced through human intervention.
3. Interoperability system allows a common perception to be disseminated to a key set of decision makers.
4. If interoperability system is implemented properly, it can potentially provide additional flexibility and adaptability into the system (e.g., enabling the interconnection of selected systems as might be required for changing conditions).

2.3.2 Interoperability Requirements

According to (Pardo and Burke, 2008), addressing interoperability requires collaborating organizations to:

- Invest in changes to internal organizational arrangements, practices and technical resources to align with externally-agreed priorities
- Create new and renewed relationships
- Recognize and manage the challenges to network formation such as modification of legal framework and organizational infrastructure.

From our experience, a number of mechanisms are required in order to achieve interoperability between the multiple database systems. These include:

- A mechanism for describing required data (concepts ,relations, data types)
- A mechanism for finding the related data in each source database
- A mechanism for collecting the required data from its source.
- A mechanism for providing homogeneous access and integrated interface to the heterogeneous and distributed databases
- Rules and a protocol for issuing requests and receiving responses in a structured manner.
- A mechanism for handling the changes on source database structure and solving the problems that may occur as a result of these changes.

Schema changes should only be done by someone who understands all the effects, someone who would know that one or more views would also need to be changed at the same time. However we should follow a common rules or protocols in each source

information system to solve the problem of changed database structure. Some of these rules are:

1. The database in each source should be always online. Don't allow to shutdown the database excepting the time after work.
2. Don't update any table used in the virtual schema. If there is any need to do this, the administrator of source database should inform and cooperate with the main system administrator to do the changes and rebuild the part of virtual schema which is affected by these changes.
3. We can use agent technology to automate the process of handling the changes and reconstruct the virtual schemas.
4. Show a message to inform the user of the interoperability system if there is a problem in any source information system (out of service) with continues receiving data from other information system and manipulating data query.
5. Use DDL auditing in each data sources in order to capture the changes on the database structures.

2.4 Ontology

It is difficult to combine the data of different information systems or establish effective communications between them unless they share a common ontology. An ontology is a representation of (some part of) a conceptualization (Guarino and Giaretta , 1995) Where a conceptualization is a world view of some domain. It is typically conceived and/or expressed "as a set of concepts (e.g. entities, attributes, and processes), their definitions and their interrelationships" (Uschold and Gruninger, 1996) which collectively impose a structure on the domain and constrain the possible interpretations of terms. Thus, it capture is a shared understanding of a domain that is agreed between a number of agents. Such agreement facilitates accurate and effective communication of meaning. This makes activities such as interoperability, reuse and sharing possible.

An ontology could, in essence, be regarded as a specification of ontological commitment which, informally, is an agreement to use a vocabulary in a way that is consistent with respect to the terms and definitions that comprise an ontology. The idea of ontological commitment was initially employed in (Gruber, 1993) and applied to Domain Ontologies. However, it was generalized in (Guarino, 1993) applied to representation ontologies. (Guarino, 1993) argued for the use of modal logic as the means for expressing the commitments.

Ontologies and ontological commitments are essential for knowledge sharing and knowledge re-use. These concepts refer to situations where a single knowledge base contains knowledge that is needed by many agents (serving the same or different applications) and/or one agent needs access to knowledge that can only be accessed through the help (reasoning) of another agent. Knowledge sharing (e.g., the exchange of informational content) enables dynamic cooperation of agents for KM tasks. Thus, the

need for a dialogue and argumentation model.

Most of the recent ontology languages are based on XML, the resource description framework (RDF), and RDF schema (RDFS) .Although other languages have been proposed, there are no universal standards yet (Pérez-Rey, et al, 2006). In the short history of ontology languages, there are numerous examples, such as Ontolingua, LOOM, OCML, FLogic, CARIN, SHOE, XOL and OML. In general, these languages are very similar. Usually, existing services and applications can be updated when new ontology languages appear (Pérez-Rey, et al, 2006).

2.4.1 Ontology Components

An Ontology consists of a number of different components. The names of these components differ between ontologies depending on the ontology language used or background of the authors. Despite this, their core components are largely shared between different ontologies.Common components of ontology include:

• Classes (concepts)

Represent concepts, which are taken in a broad sense. For instance, in the traveling domain, concepts are: cities, villages, hotels, planes, trains, cars. Classes in ontology are usually organized in taxonomies through which inheritance mechanisms can be applied. In the frame-based knowledge representation paradigm, meta-classes can also be defined. Meta-classes are classes whose instances are classes. They usually allow for gradations of meaning, since they establish different layers of classes in the ontology where they are defined. (Corch, et al., 2006)

• Relations

Represent a type of association between concepts of the domain. Ontologies usually contain binary relations. The first argument is known as the domain of the relation, and the second argument is the range. Relations can be instantiated with knowledge from the domain. Binary relations are sometimes used to express concept attributes. Attributes are usually distinguished from relations because their range is a data type, such as string, number, etc., while the range of relations is a concept (Corch, et al., 2006)

• Axioms

Serve to model sentences that are always true. They are normally used to represent knowledge that cannot be formally defined by the other components. In addition, formal axioms are used to verify the consistency of the ontology itself or the consistency of the knowledge stored in a knowledge base. Formal axioms are very useful for inferring new knowledge. (Corch, et al., 2006)

• Instances

Individuals (instances) are the basic, "ground level" components of an ontology. The individuals in an ontology may include concrete objects such as people, animals, tables, automobiles, molecules, and planets, as well as abstract individuals such as numbers and words. An ontology need not include any individuals, but one of the general purposes of an ontology is to provide a means of classifying individuals, even if those individuals are not explicitly part of the ontology.

• Attributes

Objects in an ontology can be described by relating them to other things, typically aspects or parts. These related things are often called attributes, although they may be independent things. Each attribute can be a class or an individual. The kind of object

and the kind of attribute determine the kind of relation between them. A relation between an object and an attribute express a fact that is specific to the object to which it is related.

As an example, consider a very simple ontology of part of the animal kingdom (Figure 5). In this ontology, the domain concepts are plants and animals, herbivores and carnivores, antelopes and lions. Carnivores and herbivores are both kinds of animal, so we establish 'is-a' relationships such as 'a carnivore is-a animal' and 'a lion is-a carnivore'. The 'is-a' relation is special because it has transitive properties, allowing us to derive the fact that a lion is an animal, even though this is not stated directly. The 'eats' relation is not transitive, though following it does allow us to analyze the food chain.

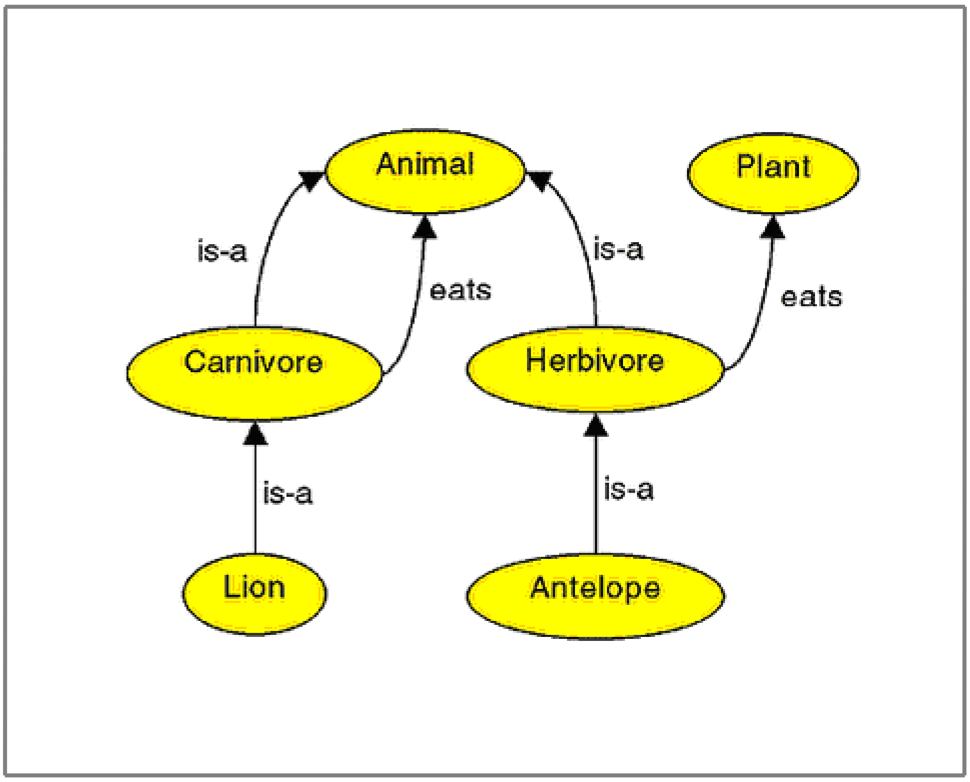


Figure 5: Part of the Animal kingdom Ontology

Another example of ontology is shown in Figure 6. In this ontology, the domain concepts are building, house, hotel, room, bedroom and bathroom, so we establish 'is-a' relationships such as ' hotel is-a building', ' house is-a building ', ' bedroom is-a room', ' bathroom is-a room '. Area is an attribute of house while color is an attribute of bathroom which can be blue, brown, etc. House and hotel are sub class of building super class while room is part of house.

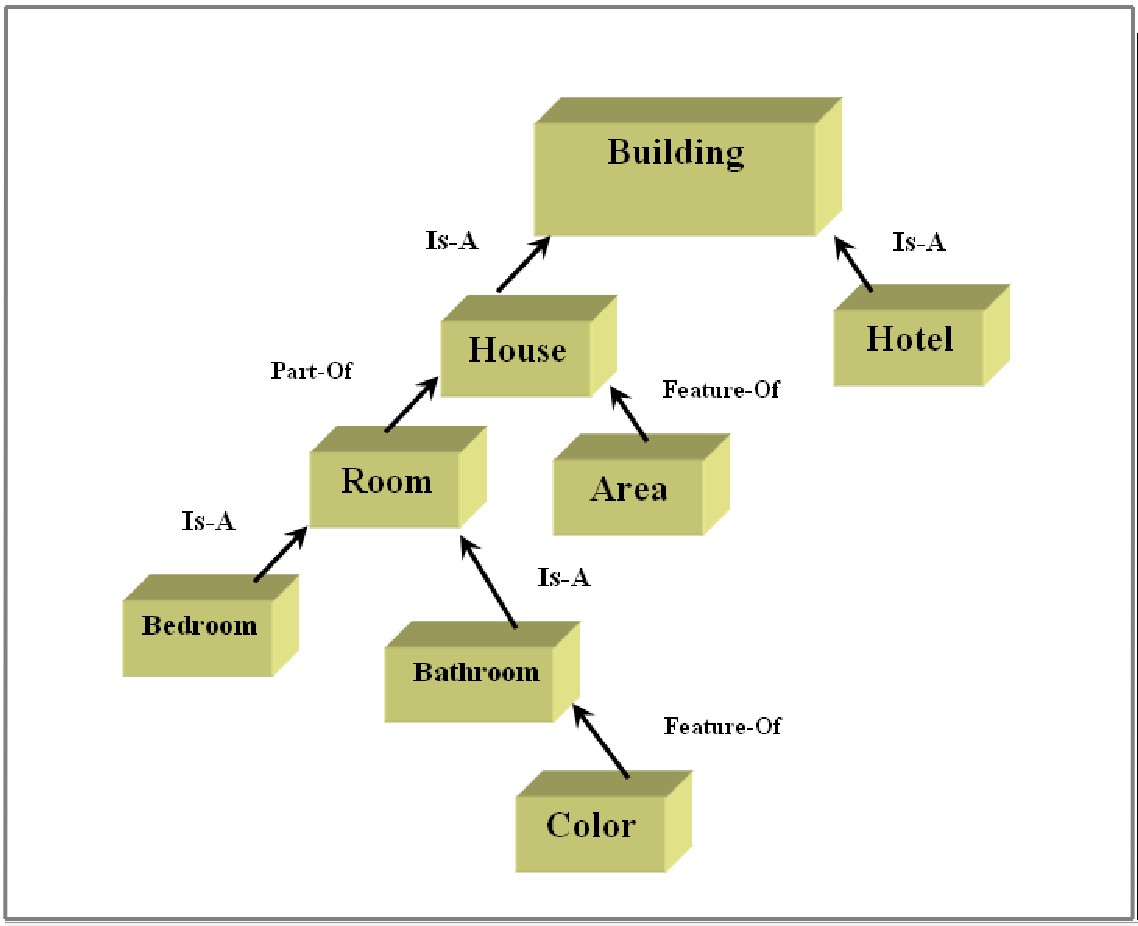


Figure 6 : Building ontology example

2.4.2 Ontology advantages

Why would someone want to develop an ontology? Some of these reasons according to (Noy and McGuinness , 2001) are:

- To share common understanding of the structure of information among people or software agents.
- To enable reuse of domain knowledge.
- To make domain assumptions explicit.
- To separate domain knowledge from the operational knowledge.
- To analyze domain knowledge.

There is no actual schema in terms of relational databases. There is a vocabulary defined; however, there are no fixed columns which have to be filled with data. New properties can be added at any time without breaking compatibility.

2.4.3 Ontology Development Lifecycle

There are no standardized methodologies for building ontologies. Such a methodology would include a set of stages that occur when building ontologies, guidelines and principles to assist in the different stages, and an ontology life-cycle which indicates the relationships among stages (Uschold and Gruninger, 1996). The most well known ontology construction guidelines were developed by Gruber (Gruber, 1993), to encourage the development of more re-usable ontologies. Recently, there has been increased effort in trying to develop a comprehensive ontology methodology (e.g. (Gruninger and Fox, 1995), (Fernandez, et al., 1997), (Uschold and Gruninger, 1996). A survey is given in (Jones, et al., 1998).In their paper, (Stevens et al, 2000) represented

two ontology development lifecycle in figure 7 and figure 8 inspired by the software engineering V-Process Model (Sommerville, 2006).

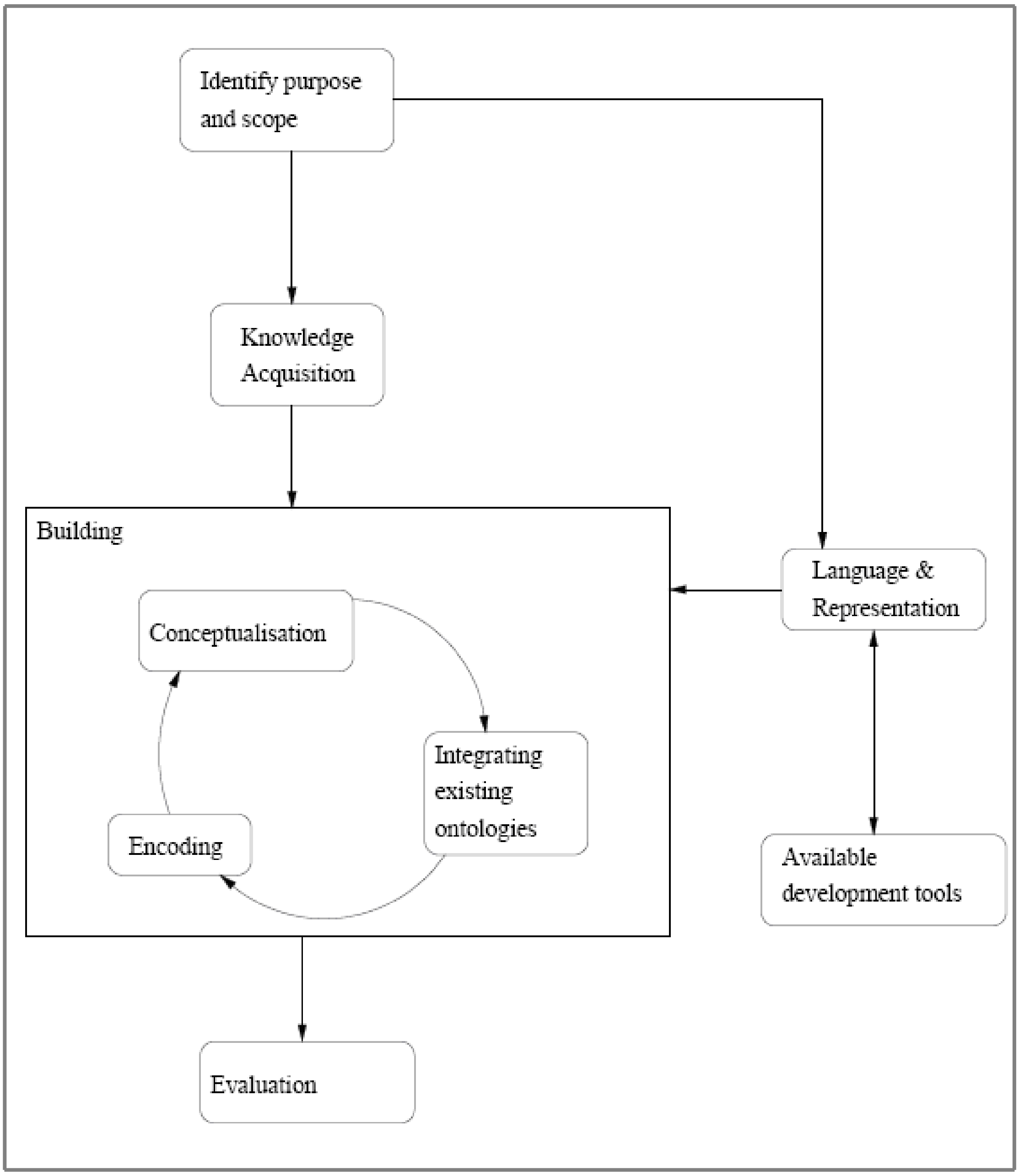


Figure 7 : Ontology Building Life-Cycle (Stevens et al, 2000)

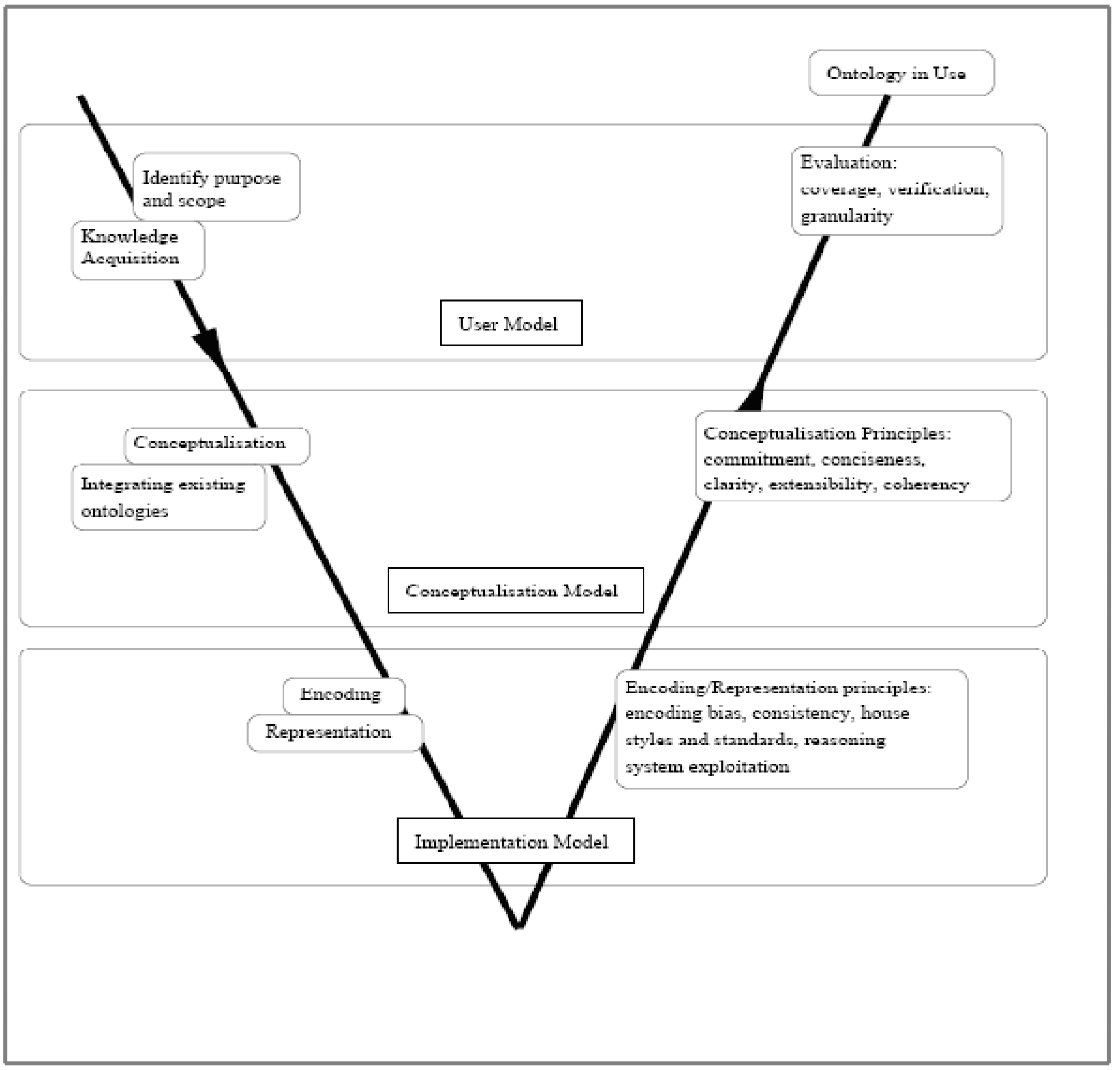


Figure 8 : The V-model Inspired Methodology for Building Ontologies (Stevens et al, 2000).

According to (Stevens et al, 2000), the stages in the V-process model and life-cycle are:

- Identify purpose and scope: developing a requirements specification for the ontology by identifying the intended scope and purpose of the ontology. A well-characterized requirements specification is important to the design, evaluation and re-use of ontology.
- Knowledge Acquisition: the process of acquiring domain knowledge from which the ontology will be built. Knowledge holders: Specialist Staff (in our case managers, social workers, psychologists, etc.) standard text books and research papers.

Informal competency questions formed, the competency questions are used to evaluate the ontological commitments that have been made to see whether the ontology meets the Requirements. (Uschold and Gruninger, 1996)

- Conceptualization: identifying the key concepts that exist in the domain, their properties and the relationships that hold between them; identifying natural language terms to refer to such concepts, relations and attributes; and structuring domain knowledge into explicit conceptual models. This is the process where the concepts and relationships describing the domain are captured. The ontology is usually described using some informal terminology. Gruber (1993) suggests writing lists of the concepts to be contained within the ontology and exploring other ontologies to re-use all or part of their conceptualizations and terminologies. At this stage it is important to bear the results of the first step, that of requirements gathering, in mind.
- Integrating: use an existing ontology: a task frequently hindered by the inadequate documentation of existing ontologies.
- Encoding: representing the conceptualization in some formal language, e.g. frames, object models or logic.
- Documentation: Documentation is important for defining, more expansively than is possible within the ontology, the exact meaning of terms within the ontology.
- Evaluation: determining the appropriateness of an ontology for its intended application. Evaluation is done pragmatically, by assessing the competency of the ontology to satisfy the requirements of its application, including determining the consistency, completeness and conciseness of an ontology.

2.5 Schema Template

The data exchange problem has received recently great attention, both from a theoretical (Fagin et al, 2005) and a practical point of view (Popa et al, 2002). The goal of data exchange is to take data structured under the source schema and transform it into a format conforming to the target schema.

Schema exchange is a natural extension of the data exchange problem in which mapping are defined over classes of similar schema (Papotti and Torlone , 2009) . The notion of schema exchange can be used to support several practical problems related to the exchange of information between heterogeneous repositories of data:

- It can be used to support the design of a data exchange setting, since a schema exchange models naturally a design pattern for data exchange, that is, a general repeatable solution to a commonly occurring data transformation from a format into another (Papotti and Torlone, 2009).
- It can support the reuse of a data exchange setting, since a schema exchange that generalizes a given data exchange mapping can be derived and later used to implement a transformation similar to the original one (Papotti and Torlone ,2009).
- A schema exchange framework can be used to represent and tackle the problem of model translation (Bernstein and Melnik ,2007), in which the goal is the translation of schemas and data between heterogeneous data models.

A graphical example of a data exchange scenario is reported in figure 9. On the left-hand side there is a source schema involving the relations ACCUSED PERSON and ACCUSED_PERS_CHARGES linked by a foreign key and, on the right-hand side, a target schema that involves the single relation PERSON_CHARGES. The

correspondences between the source and the target, expressed by means of arrows, suggest that the target can be obtained from the source by joining the relations ACCUSED PERSON and ACCUSED_PERS_CHARGES according to the foreign key, projecting the result on the attributes involved by the arrows, and storing the result into the relation PERSON_CHARGES . This transformation can be specified, in a precise way, by means of the following first order rule, called source-to-target dependency (Papotti and Torlone, 2009).

ACCUSED PERSON (~~ID~~, case_id, pers_id,hiding method ,genauth_value , military_flag , accused_status) ,
ACCUSED_PRS_CHARGE (ACSPRS_~~ID~~, CHARGE_NO)
→
PERSON_CHARGES (case_id, pers_id,hiding method ,genauth_value , military_flag , accused_status , charge_no)

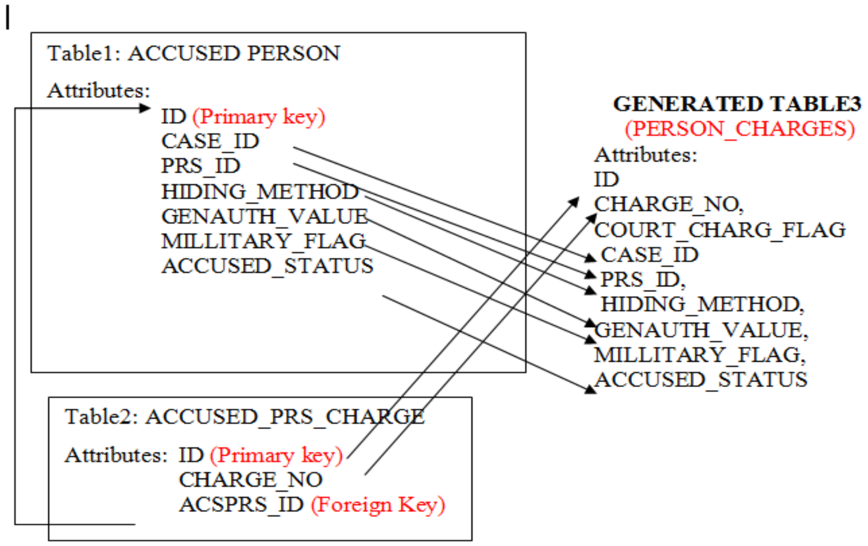


Figure 9 : An Example of Data Exchange.

The goal of the given scenario is indeed to derive in an automatic way the queries needed to translate data structured under the source schema into a format conforming to the target schema, according to the given source-to-target dependencies.

There are many different examples (consist of different tables, attribute and keys) but the underlying transformation to obtain the target is indeed similar: here, we need to ‘‘denormalize” the source by joining two relations using a referential constraint between them.

While the data exchange process operates over specific source and target schemas, the goal of schema exchange is rather the definition of generic transformations of data under structurally similar schemas. Papotti and Torlone (2009) propose a framework to schema exchange, reported graphically in figure 10. The basic idea of their approach is that this transformation can be conveniently represented, in general terms, by means of a framework, which they have called schema exchange. In this scenario we have templates of schema, that is, generic structures representing a set of database schemas with a similar configuration, and a mapping between them, informally represented by means of arrows. The aim is to represent a generic transformation that can be applied to different, but structurally similar, data sources.

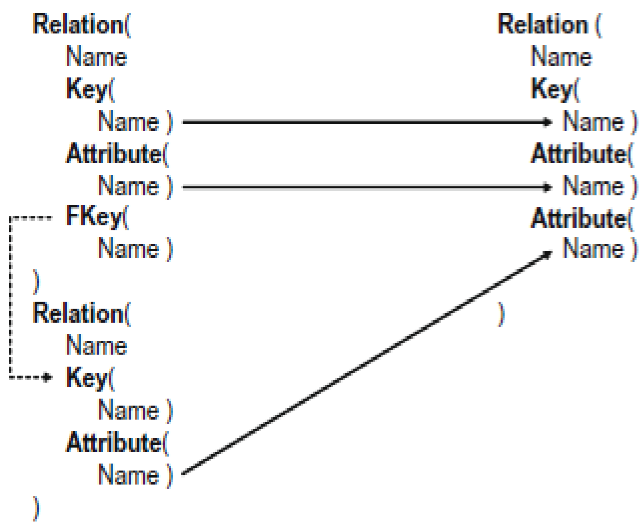


Figure 10 : An Example of Schema Exchange

3 Database Integration Approaches

3.1 Introduction

Recently, we observe the widespread usage of computer information systems in many companies and organizations. As the number of data providers and amount of data increases, integration and interoperability techniques are attracting attention. Data integration refers to combining data whereas interoperability refers to interaction between information systems and databases from different sources. Large companies tend to reduce their investment for producing data by integrating outsourced data so when creating new business units, their new applications will use existing data from various data stores, rather than new data entering the organization. Henceforth, the ability to make data stores interoperable becomes a crucial factor for the development of new information systems.

Integration of multiple database systems aims to provide users with a unified view of these data and give users the illusion of interaction with single database system. The integration process is a very complex task and must be tested from a variety of perspectives to ensure that the integration process is meeting the user requirements. There are many criteria that can be used to determine if the database integration approaches is suitable for the user needs or not. Most of these criteria will be explained in section 3.2

Currently, there are more and more approaches being developed to achieve databases integration, we can classify them into syntactic approaches like data warehouse approach, mediation approach and information linkage approach and semantic approaches like ontology based approach .The main features, advantages, and disadvantages of these approaches are explained in section 3.3

3.2 Integration Criteria

The integration criteria measure how well the database integration approach is designed, how it fitness for the purpose of integration process, and how it conforms to customer expectations. The integration criteria that we have identified are performance, cost, scalability, side-effects, reliability, encapsulation, administration, data access, and data synthesis.

- Performance: measure the execution speed of the integration technique. It examines how quickly and accurate queries, updates and other data operations are executed.
- Cost: focuses on the cost of hardware, networking, efforts and software development costs for new software or modifying existing software. The cost depends on the integration technique that will be used; some techniques need a high quality of network equipments as agent based integration approaches, while other techniques as data warehouse technique do not.
- Scalability: evaluates how well an integration technique performs as the number of databases being integrated increases.
- The side-effects: it is used to determine the changes that may be required in order to apply a specific integration technique. Some techniques require such modification on existing applications or in the computer environment as using advanced programming language.
- Security: Ability to protect data against unauthorized access and to withstand attacks that can be anticipated in its intended environment. We should have security mechanisms (such as authentication, access control and encryption) that

are adequate and correctly implemented especially when the data is very critical as criminal data.

- The data synthesis criterion examines how users work with the data from the multiple databases. The term data synthesis is used for those operations that combine data from multiple databases. Combining data can include statistical operations, calculations, joins and comparisons. This criterion is used to identify where those operations occur. For example, the data synthesis can occur in a database, in an application, or in middleware.
- Availability: focuses on readiness for correct service. In database integration systems the main goal is to get correct data from multiple resources in unified way. Depending on the integration technique, there can be time delays in making the integrated data available to users. The duration of the time delay can range from as short as minutes or hours to as long as weeks or months. From some users and applications, the time delay is a critical issue.
- Reliability: continuity of correct service
- The encapsulation criterion considers how well the integration solution hides the physical location of the data from the users and the applications. Poor encapsulation makes it more difficult for the users to use the database application. Poor encapsulation creates dependencies in the applications. The application has to know the location of the data to work with it and has to know the business rules and constraints that apply to working with that data. In the long run, these dependencies increase the applications software development /maintenance costs and make database administration more difficult. The better the encapsulation, the easier the administration. In addition to the impact of applications on database administration, the administration criterion also looks

at the impact an integration technique can have on other database administration concerns such as backup and recovery or security.

- The data access criterion focuses on what kind of access the integration solution implements for the users and applications. Data access includes create, read, update and delete access.

3.3 Integration Approaches

Database integration has been a very active research area recently. Many approaches have been proposed. Comprehensive review of these approaches is beyond the scoop of this thesis. We shall look at two different types of approaches: syntactic based and semantic based (ontology based) .The most important syntactic techniques are summarized in Table 1. The main features, advantages, and disadvantages of these approaches will be explained in detail. Finally, we review project in domain of semantic interoperability based on ontologies.

3.3.1 Data Warehouse Approach

A data warehouse is a system that extracts, cleans, conforms, and delivers source data into a dimensional data store and then supports and implements querying and analysis for the purpose of decision making (Kimball and Caserta, 2004)

This approach eliminates various problems including network bottlenecks, slow response times, and the occasional unavailability of sources.

System design using data warehouse approach is much simpler. For example, there is no need to perform query optimization over heterogeneous sources. Since query execution does not involve data translation and communication with remote sources, complex queries can be executed easily and efficiently which mean the user will have

fast data access time. Information at the warehouse is under the control of the end-users; it can be stored safely and reliably for as long as necessary. Another benefit in this approach is that it allows values (e.g., filtering, validation, correction, and annotation) to be added to the data collected from individual sources. The approach, however, suffers from the maintenance problem in light of evolution of the source database (both in structure and content). Before data can be stored within the warehouse, it must be cleaned, loaded, or extracted. These processes are costly and can take a long period of time (Widom, 1995). Compatibility problem is also one of serious problems that can be faced. If the data warehouse can be accessed via the internet, this could lead to a large number of security problems. another problem arises when the data are not updated and beside that this data need to converted in some specific format before it can be store to the warehouse, so warehouse needs to be periodically updated in order to keep a faithful image of the data which it represents, and reflect the modifications of the source databases.

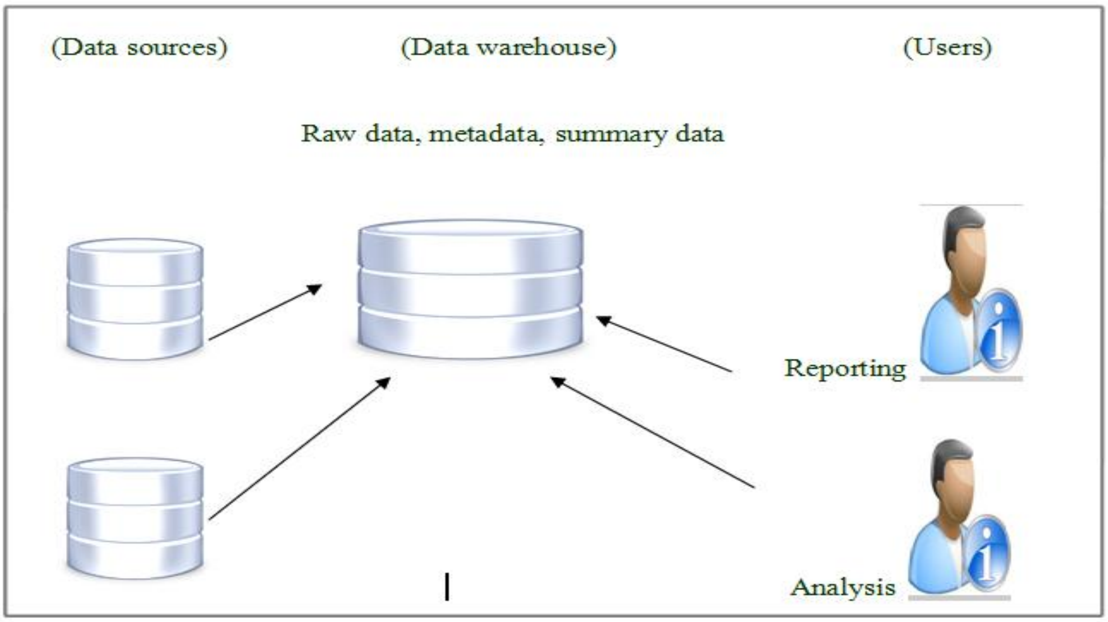


Figure 11: Simple Data Warehouse Architecture

The data warehouse is built under a very different life cycle than the classical operational environment. The data warehouse is built iteratively - quickly and in small iterations (Inmon, 2005).

Inmon(2005) determined the steps of data warehouse development which include such activities as:

- Data modeling
- Selecting hardware and software platforms
- Sizing the data warehouse
- Collecting obvious and known informational requirements
- Determining meaningful subsets of data for initial loading into the data warehouse
- Selecting the most fruitful functional area for population
- Selecting a subject area for design
- Identifying physical characteristics of the subject area
- Identifying the unit of time for the data.
- Identifying the system of record.
- Determining whether delta data should be used.
- Loading transformation data to metadata.
- Specifying the frequency of transformation.
- Executing DDL.
- Creating code that embodies the transformations.
- Allocating space.
- Population of data into the warehouse.
- Turning the end user loose on the data.

3.3.2 Mediator Approach

This approach is using mediator as a medium to integrate the data. It does not store any data, but using mediator to integrate various sources depends on user queries before display result to the user. This approach will provide various data from various sources in real time format.

The mediator is an intermediate virtual database that established between the data sources and the application using them.

As shown in figure 12 there are general mediation steps in mediation approach which start when it accepts a query and determine the appropriate set of information sources to answer the query. After that, mediator decomposes the query into sub-queries for each information source. Wrapper plays importance role to communicate with data sources and can be considered as a bridge between mediator and data source. The wrapper is used to translate the sub-queries to source specific query language. After data source produce the result, wrapper fetched the result and send back to the mediator.

Generally, wrappers are computer programs that transform data from a representation format into another format. Next, translate and filter the results into meaningful version and merge them together. After the processes are complete, final result will return to the user. There are two types of mediation systems: a global schema as view on local schema as GAV (Global As View) and local views as view of the global schema (Local as View). In LAV, local is referring to local sources or databases which mean each of local sources can be view on the global schema. An advantage of this model is easier than GAV in adding new information sources however it has low query performance. Because of that, it is better for dynamics and distributed systems.

GAV This model is a view over the source database and the advantage is that the query processing is simpler than LAV model. However, unlike LAV, there is difficult to add

new information source to this model. It is suitable for a system that consistence in used of their sources.

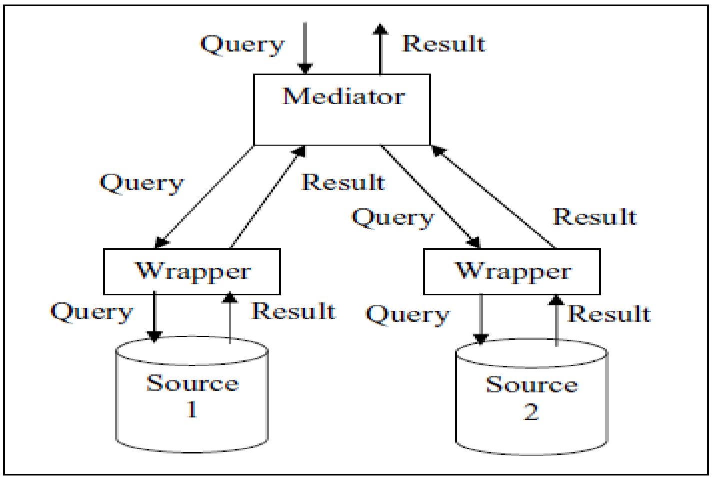


Figure 12 : General Architecture of Mediation Approach

3.3.3 Information linkage Approach

This is the simplest approach to database integration. All the related information pieces (web pages, records of databases, etc.) are linked together using static links that can be followed by users.

Although this approach is not time demanding in terms of development and simplicity, it has severe drawbacks. For instance, many of the links often are unidirectional. Users cannot go forward and back easily when they want. Regarding queries, in most cases users cannot define their own queries. Queries are limited or predefined by the static links established when the system was developed. This issue may represent an important limitation. At present, almost all the systems based on this approach are web-based provided the huge success and wide dissemination of the World Wide Web (WWW). Many are built by using some kind of hypertext language as HTML, XML. Some examples of the most important web-based biomedical databases, available over Internet: MEDLINE, PDB, PROSITE

Table 1: Comparison between Database Integration Approaches

Method	Main features	Advantages	Disadvantages
Data Warehouse Approach	Data from the remote heterogeneous database are copied on a local server and the user will use a unique interface within the system to allow multi-database queries to be issued to this single interface. -Subject oriented which means, information is presented according to specific subjects or areas of interest. - uses high-level query language like the SQL	- End users can use a single data model and query language - high performance because query optimization can be performed locally - Reliable because of it fewer dependencies on network connectivity. -users will be able to access a large amount of information -Fast access	- the warehouse must be updated regularly -The stored data has to be checked for redundancy, cleaned and loaded - Huge scale and expensive cost. -End-user training: Users of data warehouse require training to utilize all facilities provided by the repository.
Mediator approach	This approach provides a virtual view of the integrated sources. It translate query from the user, into query that is understood by the sources integrated into the system. It maps the relationship between source descriptions and the mediator and thus allows queries on the mediator to be translated to queries on the data source. - on-demand approach - Best approach to integrate multiple sources that contain dynamic data. - provide various data from various sources in real time format	- Cheaper and effective approach since it involves schema or view integration, rather than to have huge storage to store copied data from all the involved data sources. - the query directly goes to the original source so we not have the update problem	- network bottleneck - needs more time to retrieve the data because this approach needs more time according to connectivity level and how complex the query are. - Redundancy or duplication can happen if there is no method used in mediator to combine and filter the result.

Method	Main features	Advantages	Disadvantages
Information linkage approach	The system will provide static links between data or records in different data sources that can be followed by user.	It is popular because of its easy to use feature and it only involves the tasks of point and click. This approach also allows the representation of cases where the page containing the desired information is only reachable through particular navigation path across other pages	- User must specify which data sources should be used to answer a given query. - when a user is interested in a join between two data sources in the system, the user must manually perform the join by clicking on each entry in the first data

3.3.4 Ontology Based Approaches

An important potential solution to semantic heterogeneity problems, which has been attracting attention, is an ontology. This has led many researchers to use ontologies in their work (Guarino, 1998), (Welty and Smith, 2001). Many sets of criteria have been used for the classification of existing ontology-based information integration systems (Wache, et al. , 2001), (Bellatreche,et al .,2006) . One sample criterion is the place where information resides, which allows us to distinguish between mediator and warehouse approaches (Ullman, 1997). Another example is the distinction between systems using a single ontology, multiple ontologies, and hybrid approaches with shared and non-shared ontologies. Others work focus on the degree of automation of mappings between sources and ontologies (Bellatreche,et al. ,2006).

Carnot (Singh, et al., 1997) is one of the pioneer projects in the domain of integration and addresses semantic integrity. InfoSleuth is a multiagent-based solution for information and service retrieval from autonomous and changing sources such as those

on the Internet. Its architecture relies on different types of agents such as user agents, ontology agents, query agents and so on. A user can select an ontology (via a user agent) from a list of ontologies offered by an ontology agent. The ontologies are built for every user group (Fowler, et al., 1999) and InfoSleuth offers the ability to build queries based on a selected ontology and consequently, it matches information or services with the user queries. . ONTOFUSION (Pérez-Rey, et al., 2006) system is an ontology-based system designed for biomedical database integration. It is based on two processes: mapping and unification. Mapping is a semi-automated process that uses ontologies to link a database schema with a conceptual framework—named virtual schema. There are three methodologies for creating virtual schemas, according to the origin of the domain ontology used: (1) top-down—e.g. using an existing ontology, such as the UMLS or Gene Ontology—, (2) bottom-up—building a new domain ontology— and (3) a hybrid combination. Unification is an automated process for integrating ontologies and hence the database to which they are linked. Using these methods, they employed ONTOFUSION to integrate a large number of public genomic and clinical databases, as well as biomedical ontologies. ONTOFUSION presents a suitable architecture for semantic interoperability and inspired parts of work presented in this thesis.

KRAFT (Visser ,et al., 1999) is a project for the integration of heterogeneous information, using ontologies to resolve semantics problems. They extract the vocabulary of the community and the definition of terms from documents existing in an application domain.

4 Our Integrative Approach to Database Interoperability

We propose a new hybrid approach for implementing interoperable database system based on ontology and schema template to integrate multiple heterogeneous data sources that contain related data .As described in figure13, we start building a domain ontology by understanding at least two related databases (Mizan and police databases in our project) and finding the best vocabularies to represent the domain concepts, attributes and relations. We also used the notion of creating ontology from databases (Cullot and Ghawi, 2007) to create the ontology. In each data source, we built a template schema that contains table's names, attributes, keys and relations. Specific rules that defined with our template will facilitate the process of adding new data sources. We can also use the template to track the changes on source data structures. After that, we built a virtual schema in each data source, which is a conceptual ontology. Concepts, relationships and attributes for virtual schema are picked from domain ontology which is created previously. Afterwards, every element in the virtual schema is mapped to a respective element in the physical schema. In unification process, multiple virtual schemas corresponding to different databases are merged into a unified virtual schema. Then the graphical user interface is built using the unification schema.

The main steps of our approach are:

1. Ontology Creation
2. Template Creation
3. Virtual Schema Creation
4. Unification schema creation
5. Schedule Creation to track the changes on database structure
6. GUI (Graphical User Interface) creation

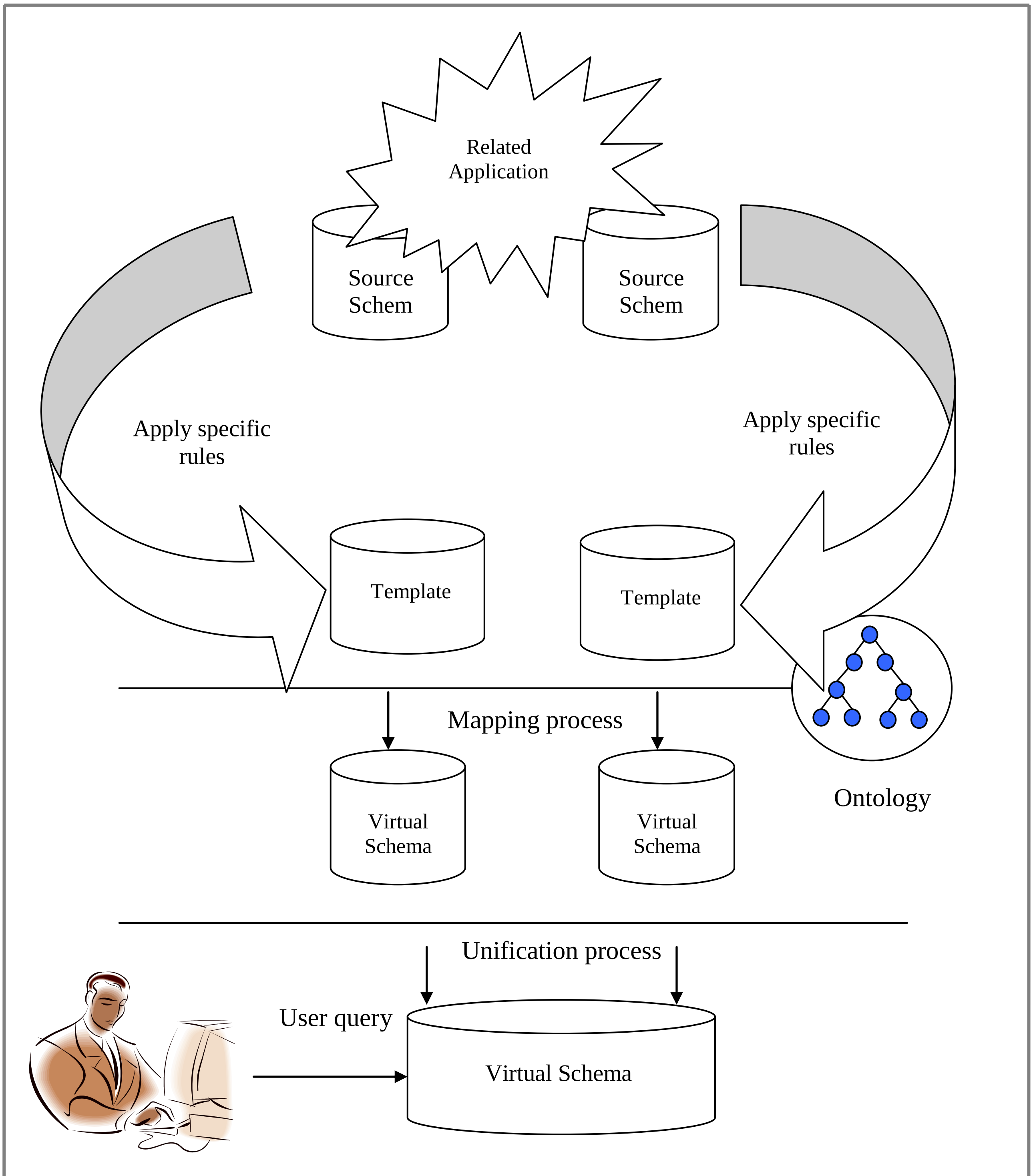


Figure 13 : The Proposed Approach to Database Interoperability.

.

4.1 Ontology Creation

The first process in our interoperability database system is to create or discover an appropriate ontology for the domain with general concepts, attributes and relations. The important question that arises is how to obtain the ontology?

Domain ontology is defined as conceptual hierarchies with general concepts, attributes and relations. The ontology is built by understanding and analyzing the domain which we want to create the ontology for it. We can also use other domain ontologies and progressively add a new concept to it whenever a new concept has been appeared and has not yet been incorporated.

The purpose of domain ontology is to guarantee that only generally accepted concept names are used to describe the different elements in virtual schemas that will be built in each data source in our integration system.

There are many methodologies are used to create the ontology, one of these methodology is "create ontology from a database" (Cullot and Ghawi, 2007) as shown in figure 14. By this methodology, we convert the database tables to concepts in ontology and covert database columns to attributes that describe these concepts, whereas primary keys and foreign keys used to found the relations between the concepts.

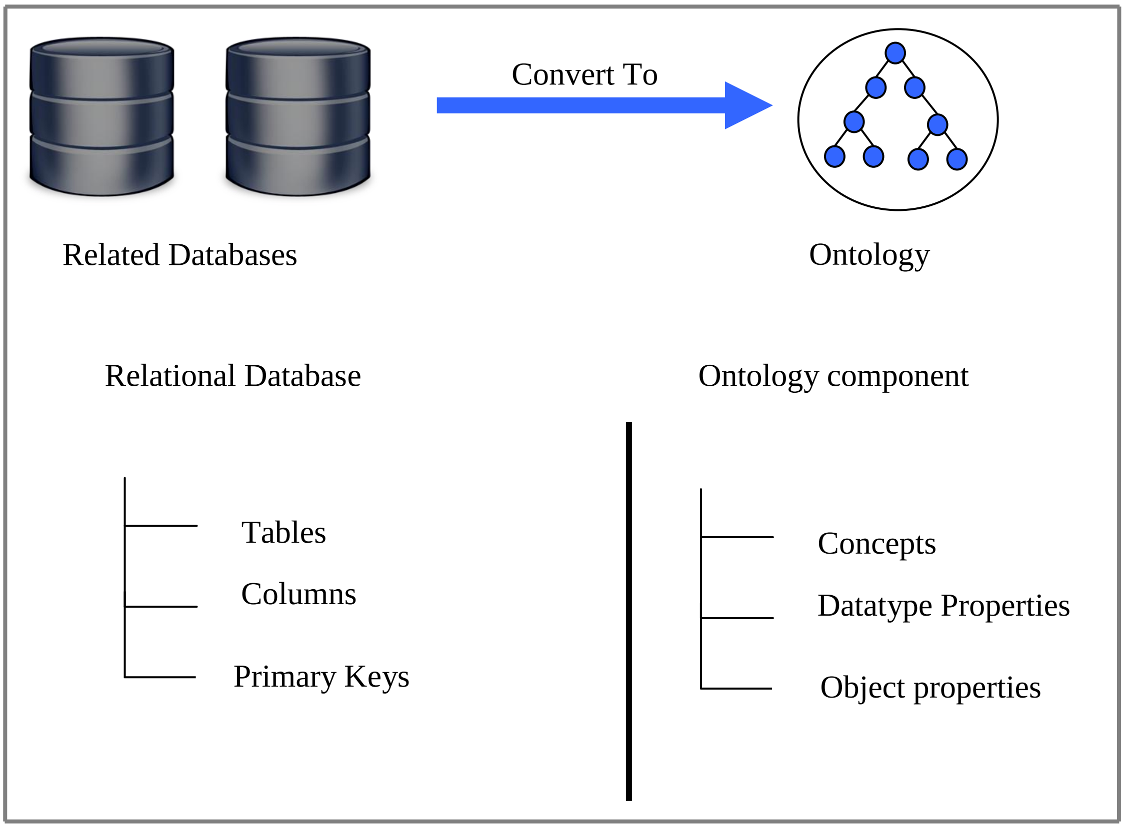


Figure 14: The Relationship between Ontology Components and Relational Database Components

Developing an ontology includes:

1. Defining classes in the ontology. In our legal domain , the main classes are shown in figure 15 :

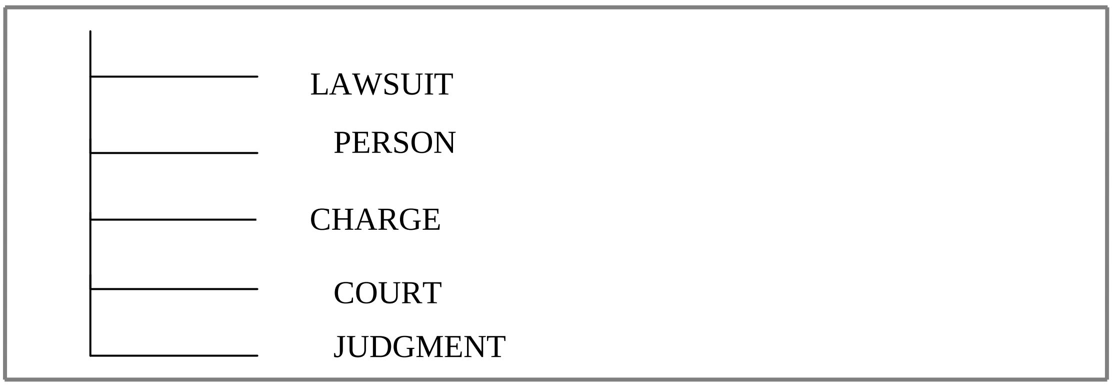


Figure 15 : Our System Ontology Classes

2. Arranging the classes in a taxonomic (subclass–super class) hierarchy as shown in figure 16.

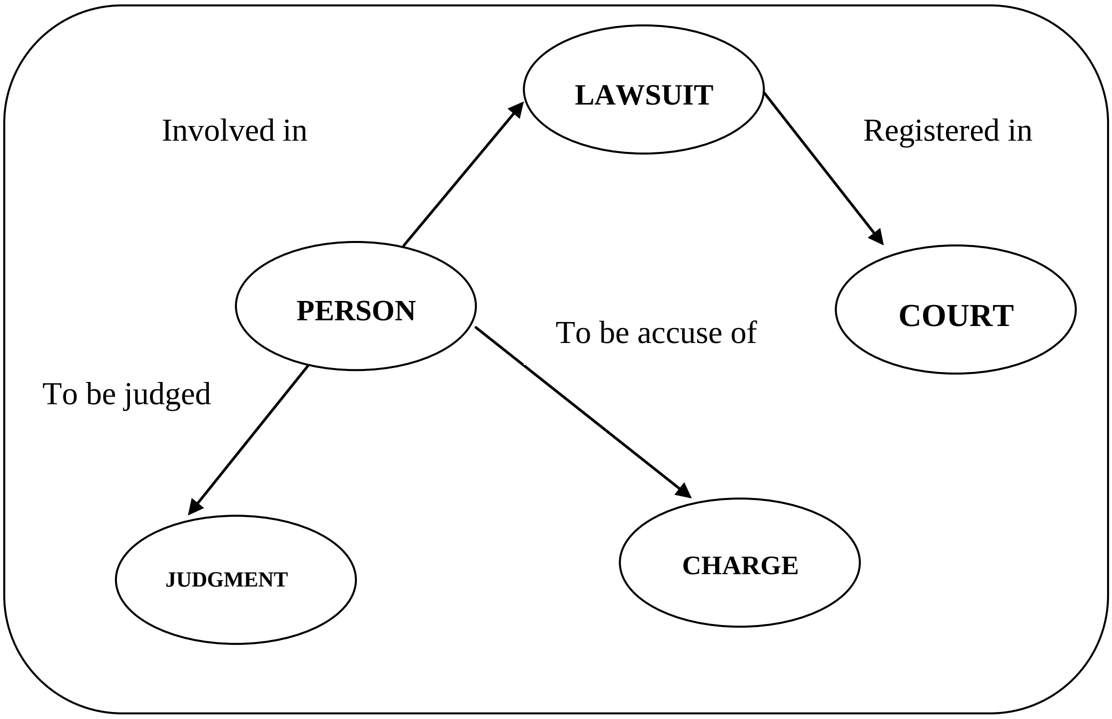


Figure 16 : Our System Ontology Hierarchy

3. Defining attributes and describing allowed values for these attributes for each class that is defined in the previous step as shown in the following tables.

Table 2 : The Attributes of Lawsuit class and its Allowed Values

LAWSUIT ATTRIBUTES	
ATTRIBUTE NAME	ALLOWED VALUE
LAWSUIT_ID	(NUMBER)
LAWSUIT_DATE	(DATE)
LAWSUIT_STATUS	(ENUMERATED)
LAWSUIT_CLASS_CODE	(ENUMERATED)

Table 3 : The Attributes of Person class and its Allowed Values

PERSON ATTRIBUTES	
ATTRIBUTE NAME	ALLOWED VALUE
NATIONAL NUMBER	STRING
FULL NAME	STRING
GENDER	ENUMERATED (MALE, FEMALE)
BIRTH DATE	DATE
JOB	STRING
PERSON TYPE	ENUMERATED
BIRTH PLACE	ENUMERATED

Table 4 : The Attributes of Charge class and its Allowed Values

CHARGE ATTRIBUTES	
ATTRIBUTE NAME	ALLOWED VALUE
CHARGE NUMBER	NUMBER
CHARGE_NAME	STRING
CHARGE CLASSIFICATION	ENUMERATED

Table 5 : The Attributes of Court class and its Allowed Values

COURT ATTRIBUTES	
ATTRIBUTE NAME	ALLOWED VALUE
COURT_CODE	NUMBER
COURT_NAME	STRING
COURT LOCATION	ENUMERATED
COURT CLASSIFICATION	ENUMERATED

Table 6 : The Attributes of Judgment class and its Allowed Values

JUDGMENT ATTRIBUTES	
ATTRIBUTE NAME	ALLOWED VALUE
JUDGMENT NUMBER	NUMBER
JUDGMENT_STATMENT	STRING
DECISION_DESCRIPTION	ENUMERATED

4.2 Virtual Schema Creation

In each data source, we built a virtual schema which is a conceptual ontology. Concepts, relationships and attributes for virtual schema are picked from domain ontology which is created in the previous step. Ontologies ensure that all semantically equivalent elements in different schemas use the same concept name (Pérez-Rey, et al., 2006). When a new virtual schema is being developed, concept names cannot be chosen arbitrarily. They should be selected from the respective domain ontology. In this sense, domain ontologies provide the semantic links between different schemas, which are needed to carry out the schema unification process later.

In figure 17, the mapping process is carried out as follows. First, a new ontology is created by finding the most suitable representation of the domain concepts and relationships. Then in each data source the virtual schema is built by selecting classes, relationships and properties from the domain ontology. Afterwards, every element in the virtual schema is mapped to a respective element in the physical schema.

This step was accomplished by understanding the source schema, tables, attributes and relationships between tables which contain the required data in order to be able to map the elements in the virtual schema to the respective elements in the source schema. (See appendix C)

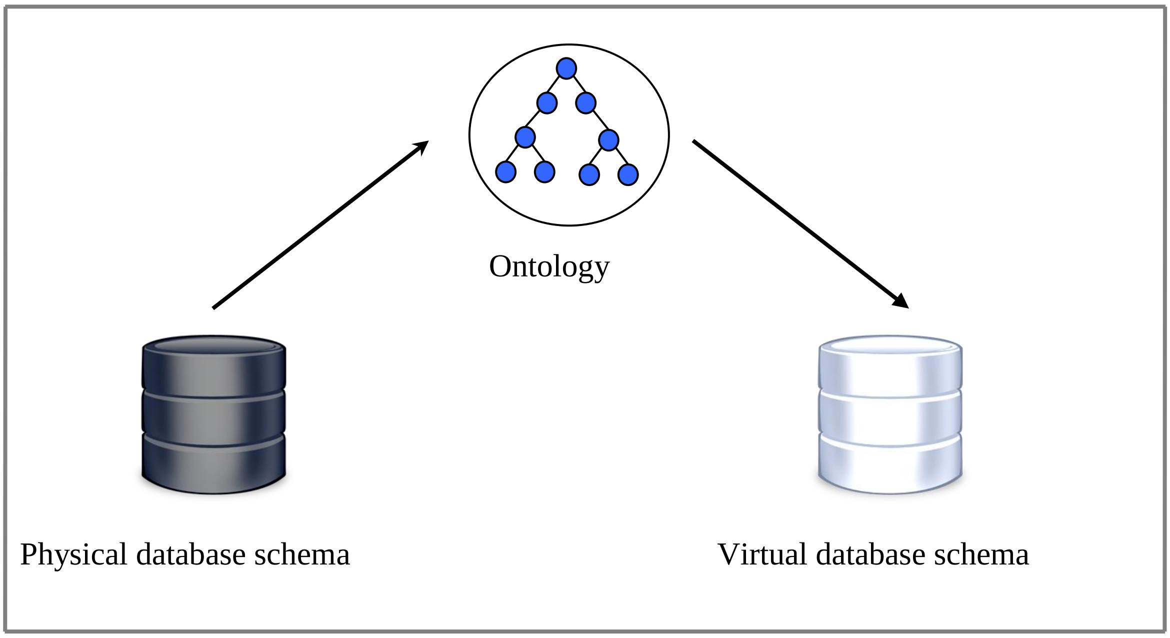


Figure 17: Mapping Process

However, when we executed some queries, we noticed that the information about the same person from different data sources is conflicting and incompatible. In order to have accurate and up to date information about any citizen like full name, birth date, religion and marital status, we used a new data source that have the correct information about any person in the country, which is civil status system. we used the civil status system to retrieve the required information about any citizen and make this data source interoperable with other databases (Mizan system and police system). So when such changes occur in the civil status, we will get it.

4.3 Template Creation

In each database sources (Mizan and police), we built a template that contains table's names, attributes, keys and relations before the interoperability system was activated (see Appendix D) The template can be used to capture the changes on database structure by comparing this template with the database structure at a specific time. The goal of template is to track the changes on database schema at any time. These scripts are used to build a template that contain table's names, attributes, keys and relations which have all needed data without concern to other tables , so when any table structure - not related to our project –is changed , our system will not be affected.

TABLES
NAME
CASES
ACCUSED PERSON
ACCUSED_PRS_CHARGE
JUD_SENTENCES
SYS_PERSONS

KEYS	
NAME	IN
ID	CASES
ID	ACCUSED PERSON
ID	ACCUSED_PRS_CHARGE
ID	JUD_SENTENCES
ID	SYS_PERSONS

ATTRIBUTES		
NAME	NULLABLE	IN
FILE_NO	NOT NULL	CASES
NATIONAL_NO	NOT NULL	SYS_PERSONS
ACCUSED_STATUS	NULL	ACCUSED PERSON
CHARG_NO	NOT NULL	ACCUSED_PRS_CHARGE
BAIL_AMOUNT	NULL	JUD_SENTENCES

FOREIGN KEYS		
NAME	IN	REFERENCE
CASE_ID	ACCUSED PERSON	CASES(ID)
PERS_ID	ACCUSED PERSON	SYS_PERSONS(ID)
ACSPRS_ID	ACCUSED_PRS_CHARGE	ACCUSED PERSON(ID)
ACSPRS_ID	JUD_SENTENCES	ACCUSED PERSON(ID)

Figure 18 : Template Example

4.4 Schedual Creation To Handle The Source Schema Changes

We built and run DBMS_JOB in oracle 10g to compare the current table's structure with the template which was built previously. This job can be executed when an error is occurred while data retrieving process or when the administrator wants to do that. If any different is found the system administrator will call the database owner and cooperate with him to solve the problem and reconstruct the part of virtual schema which was affected. Then the template will be rebuilt, thus we have a new copy of template according to the new changes.

Another way to handle the changes that may be occurred is building (data definition Language) DDL triggers, (see appendix E)

4.5 Unification Process

In a unification process multiple virtual schemas corresponding to different databases are merged into a unified virtual schema. (See appendix F). Unified virtual schemas are ontologies that reflect the conceptual structure of the information stored in various databases (Pérez-Rey D, et al., 2006). Thus, they act as descriptions of virtual databases, matching “real” data from physical databases. Mapping process and unification process are working together as shown in figure 19.

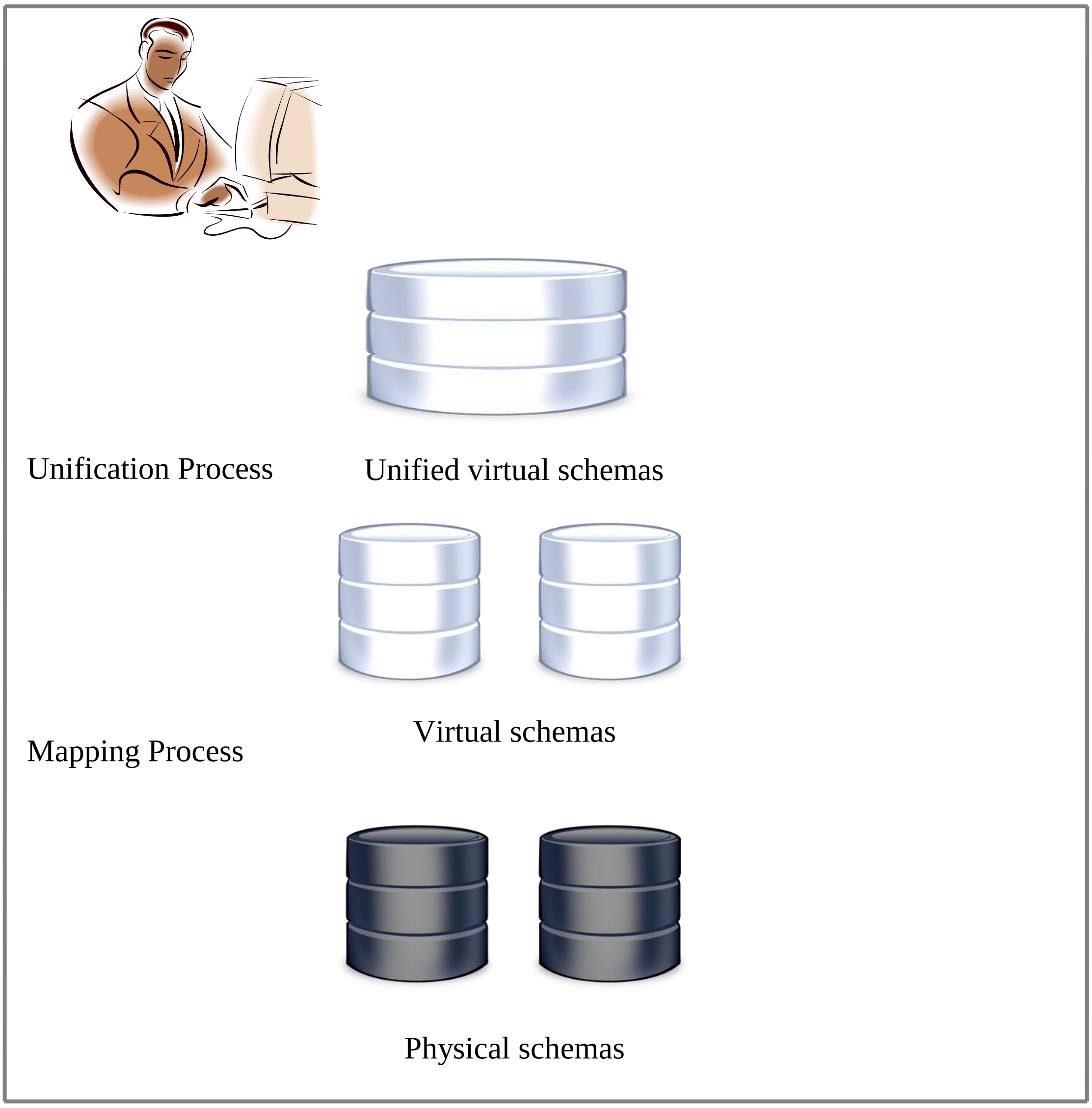


Figure 19: Unification and Mapping Processes

4.6 Graphical User Interface(GUI) Creation

We built a Graphical User Interface (GUI) using the unified virtual schema that built previously to help the users to access the heterogeneous databases (Mizan database and police database) in a unified way. The GUI is designed to be more flexible especially when one of the sources database is down or one of the source schema structure is altered. Therefore, the GUI gives the users multiple options to retrieve the data from a specific source without other or to retrieve the data from all the sources. Figure 20 represents the main screen in our system. It allows us to determine the source of our data. The user can press the button on the right side to retrieve the data from Mizan and Police information systems at the same time. The second choice enables the user to get the required data from only the courts in the ministry of justice as shown in figure 22. The last button, which is on the left side, retrieves the data from only the police information system as shown in figure 23.



Figure 20: Interoperability System.

Figure 21 represents our Criminal CV Interoperability System, it allows heterogeneous systems (Mizan, police and civil status information systems) to interact, interoperate and exchange information in a meaningful way. The user should enter the national number of the citizen whom he wants to know his criminal CV. the citizen personal data will be appeared in the upper section of the screen. The personal data is fetching from the civil status information system. Then all the citizen lawsuits and their related data as data source, lawsuit number, court name and lawsuit status will be listed. The next section of the screen shows the citizen charges in each lawsuit. The last section of the screen shows the judgments if there exist.



Figure 21: Criminal C.V Interoperability System.



Figure 22 : Criminal C.V in ministry of justice.

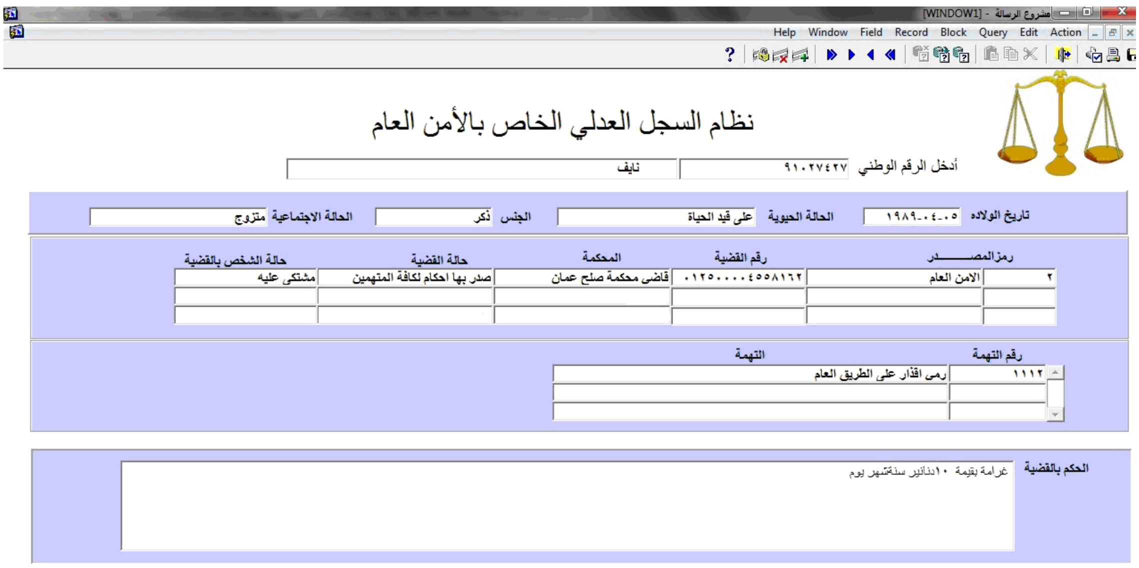


Figure 23: Criminal C.V system in public security.

5 Conclusion and Future Work

In this thesis, we presented a new hybrid approach for implementing interoperable database system based on ontology and schema template to integrate multiple heterogeneous data sources that contain related data. We built a template schema that contains table's names, attributes, keys and relations in the early stage of building our system. The template can enable the system administrator to capture the changes on database structure by comparing this template with the current database structure. The goal of template is to track the changes on database schema at any time.

We create our ontology from understanding several database schemas (Mizan and Police) that relate to the same domain. Ontology is capable of solving the problem of semantic heterogeneity and avoiding ambiguousness. We used the created ontology to build a virtual and unification schema that allow us to have a unified view of these data and give us the illusion of interaction with single database system.

To evaluate the effectiveness of the proposed approach, we compared our integrated system with other systems built using other approaches. The results indicate that our approach is effective because it has tremendous potential in solving many of the problems faced by other integration system based on approaches like warehousing approach and mediation approach at the same time. Unlike these integration approaches, our approach provides users with immediate access to freshest version of each source of databases (i.e. the data is up to date) without storing any data like warehousing approach so our users like the judges can have the freshest version of the criminal CV of any participant in the lawsuits thus maximizing their ability of decision making. At the same time, based on our approach, the built system is very high scalable so we can

add many information systems like Border & residency, Family Protection and military courts to it when its virtual schema based on the common ontology is built unlike the mediation approach. Also we solve the problem of semantic heterogeneity which explained in chapter 2 by using ontology technology. We built Flexible and user friendly interface so if any source information system is out of service the system can retrieve the data from other systems. The interoperability system allows us to bridge the gaps among different data sources, databases and formats. Other organizations like state security court can use our ontology to build new system or update their system according to our ontology that related to their domain. By using the built ontology the, the organizations can eliminate time-consuming data conversion and reduce associated costs.

Our project still has some limitations. It can not solve the problem of changing the source database effectively, so any system could be out of the services until the database structure changes that affect our system are solved by the system administrator. Another limitation is the data time retrieval. This approach needs more time according to query connectivity level and how complex the queries are.

In the future we can apply agent technology to automate the process of handling the changes on source database structure immediately, reconstructing the virtual schemas according to these changes with continuing to provide the system services and inform the user that a certain source is changed and need a time to retrieve the data which mean the queried data might be exist in this source.

References

Cullot, N., Ghawi, R. (2007), **Building Ontologies from Multiple Information Sources,**

D.M. Jones, T.J.M. Bench-Capon, and P.R.S. Visser (1998), **Methodologies for Ontology Development**. In J. Cuena, editor, Proc. ITi and KNOWS Conference of the 15th IFIP World Computer Congress, pages 62.75, London, UK. Chapman and Hall Ltd.

Fowler, J., Perry, B., Nodine, M., and Bargmeyer, B. (1999). **Agent-based semantic interoperability in InfoSleuth.** SIGMOD Record, 28(1).

GUARINO, N. and GIARETTA, P. (1995), **Ontologies and knowledge bases: Towards a terminological clarification,** In Mars, N., editor, Towards Very Large Knowledge Bases: Knowledge Building and Knowledge Sharing 1995, pages 25-32, Amsterdam, NL. IOS Press.

GUARINO, N. (1995). **Formal ontology: conceptual analysis and knowledge representation**, International Journal of Human-Computer Studies, 43:625 - 640.

L. Popa, Y. Velegrakis, R.J. Miller, M.A. Hernndez and R. Fagin (2002) ,**Translating web data**, in: International Conference on Very Large Data Bases (VLDB),pp. 598–609.

M. Fernandez, A. Gomez-Perez and N. Juristo (1997), **METHODONTOLOGY: From Ontological Art to Ontological Engineering**, In Spring Symposium Series, pages 33.40, Menlow Park, California, AAAI Press.

M. Gruninger and M. S. Fox (1995), **Methodology for the Design and Evaluation of Ontologies**, In IJCAI Workshop on Basic Ontological Issues in Knowledge Sharing.

Mike Uschold and Michael Gruninger (1996), **Ontologies : principles, methods, and applications,** Knowledge Engineering Review, 11(2) :93–155.

Noy, N. F. and McGuinness , D. L. (2001), **Ontology Development 101: A Guide to Creating Your First Ontology**, Available at http://protege.stanford.edu/publications/ontology_development/ontology101-noy-mcguinness.html

Oscar Corch et al. (2006), **Ontological Engineering: Principles, Methods, Tools and Languages,** Springer 1-48

Pardo, T.A. and Burke, G.B. (2008) , **Improving Government Interoperability: A Capability Assessment Framework for Government Managers**, Center for Technology in Government, University at Albany, SUNY.

P.A. Bernstein and S. Melnik (2007), **Model Management 2.0: Manipulating Richer Mappings,** in: SIGMOD Conference

P. Papotti and R. Torlone (2007), **Schema exchange: Generic mappings for transforming data and metadata**, in: International Conference on Conceptual Modeling (ER)

R. Fagin, P.G. Kolaitis, R.J. Miller, L. Popa (2005) , **Data exchange: semantics and query answering,** Theoretical Computer Science 336 (1) (2005) 89–124.

Robert Stevens, Carole A. Goble and Sean Bechhofer (2000), **Ontology-based Knowledge Representation for Bioinformatics**, University of Manchester

Thomas R. Gruber (1993), **Toward Principles for the Design of Ontologies Used for Knowledge Sharing**, Stanford Knowledge Systems Laboratory

Singh, M. P., Cannata, P. E., Huhns, M. N., Jacobs, N., Ksiezyk, T., Ong, K., Sheth, A. P., Tomlinson, C., and Woelk, D. (1997). **The Carnot heterogeneous database project: Implemented applications. Distributed and Paralleled Databases**, 5(2):207–225.

William H. Inmon(2005), **Building the Data Warehouse / Edition 4**

Pérez-Rey D, Maojo V, García-Remesal M, et al , 2006, **ONTOFUSION: Ontology-based integration of genomic and clinical databases**. Comput Biol Med.

Guarino, N., editor (1998), **Formal Ontology in Information Systems**, IOS Press, Amsterdam.

Welty, C. and Smith, B. (2001), **Proceedings of the International Conference on Formal Ontology in Information Systems**, volume 2001, Ogunquit, Maine, USA. ACM/SIGART, ACM Press.

H. Wache, T. Vögele, U. Visser, H. Stuckenschmidt, G. Schuster, H. Neumann, S. Hübner(2001), **Ontology-based integration of information _ A survey of existing approaches**, in: H. Stuckenschmidt (Ed.), IJCAI_01 Workshop: Ontologies and Information Sharing, 2001, pp. 108_117.

L. Bellatreche, D.N. Xuan, G. Pierra and H. Dehainsala(2006), **Contribution of ontology based data modeling to automatic integration of electronic catalogues within engineering databases**, Computers in Industry Journal (2006) 711_724.

J.D. Ullman(1997), **Information integration using logical views**, in: ICDT'97: Proceedings of the 6th International Conference on Database Theory, Springer-Verlag, London, UK, 1997, pp. 19_40.

Jones, D. (1998). **Developing shared ontologies in multi-agent systems. In ECAI'98 Workshop on Intelligent Information Integration**, Brighton, U.K.

Brachman, R. J., McGuinness, D. L., Patel-Schneider, P. F., Resnick, L. A., and Borgida, A. (1991). **Living with classic: When and how to use a kl-one-like language.**In Sowa, J., editor, Principles of Semantic Networks: Explorations in the representation of knowledge, pages 401–456. Morgan-Kaufmann.

Mena, E., Kashyap, V., Illarramendi, A., and Sheth, A. (1998). **Domain specific ontologies for semantic information brokering on the global information infra structure**. In Guarino, N., editor, Formal Ontology in Information Systems. IOS press.

Visser, P. R., Jones, D. M., Beer, M., Bench-Capon, T., Diaz, B., and Shave, M. (1999). **Resolving ontological heterogeneity in the KRAFT project. In 10th International Conference and Workshop on Database and Expert Systems Applications DEXA'99**. University of Florence, Italy.

Kimball, R., and Caserta (2004), J, **The data warehouse ETL toolkit: practical techniques for extracting, cleaning, conforming, and delivering data,** Canada: Wiley Publishing.

Widom, J. (1995). **Research problem in data warehousing**. Paper presented at Proceeding of 4th International Conference on Information and Knowledge Management (CIKM)

http://en.wikipedia.org/wiki/Data_integration

Appendixes

Appendix A: Criminal CV Information SystSems.

MIZAN is a Court Case Management System that automates most routine clerical procedures performed by court staff from filing to disposition. Such procedures include capturing data on the parties involved in the case, scheduling hearings and preparing hearing notices, recording documents received by the court, preparing hearing minutes, judicial orders and judgments. MIZAN has replaced numerous manual, error-prone, and often redundant clerking tasks and docket books. MIZAN also produces statistical caseload reports useful to court managers and Ministry of Justice executives for allocating court resources and aiding special studies on the operations and performance of the courts. MIZAN is operational in all of the Conciliation, First Instance, Appeals, Cassation, and High courts in the Kingdom. In addition to computerization of the clerk's offices and court rooms of these courts MIZAN provides computerized functionality to various court departments including the Prosecutor's Office, Attorney General's Office, Execution of judgment Department, Notification Department, Mediation centers, and the Notary Public Department.

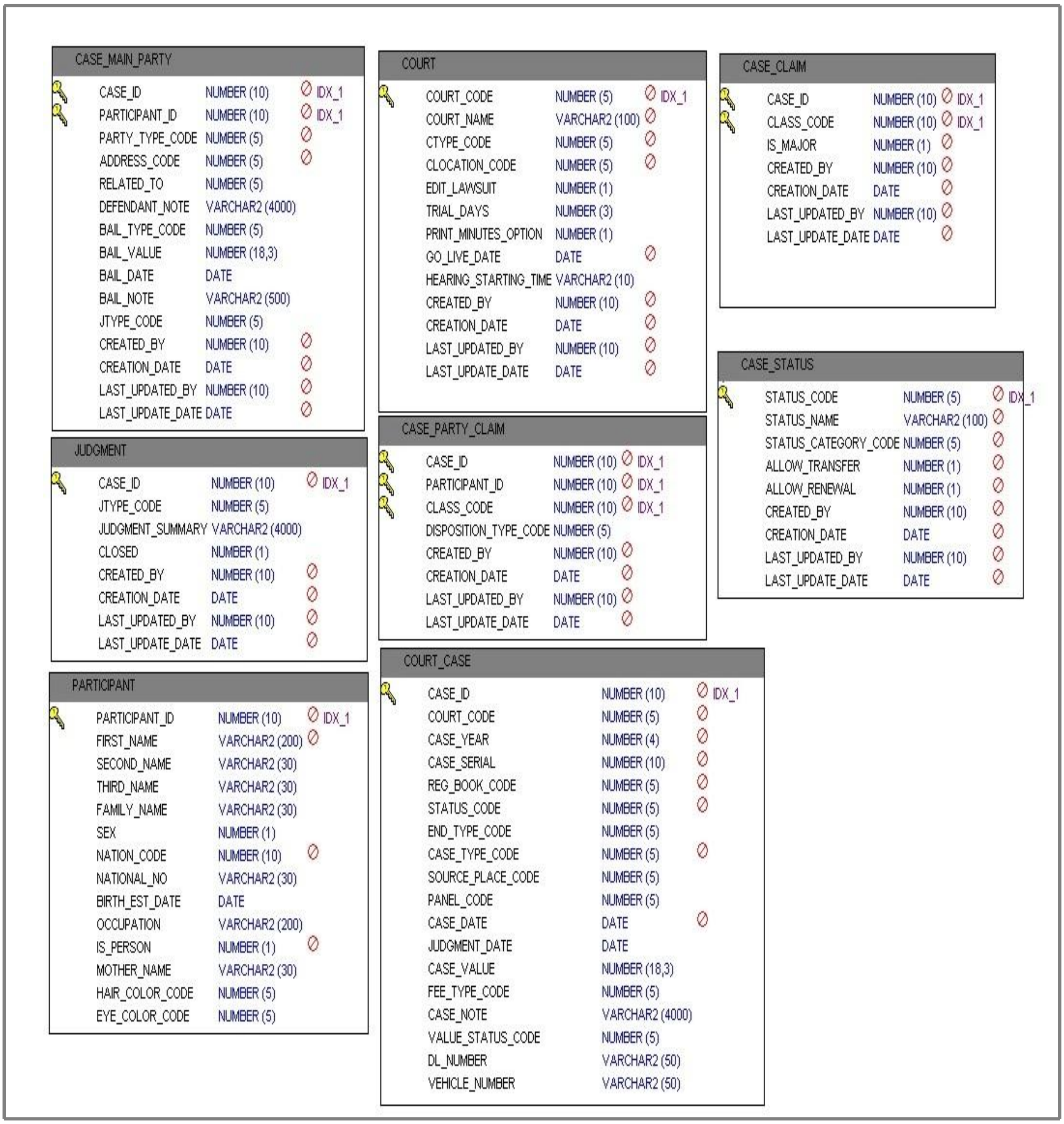
The Public security application store many type of crime category, criminal activity and other pertinent crime statistics within a particular district or in the community as a whole. All data is saved into oracle Database. You can easily find a report entered into the database by searching for the incident location, the names of involved persons.

State Security Court: It is composed of both military and civilian judges. Its jurisdiction includes offenses against the external and internal security of the state as well as drug-related and other offenses. The findings of this court are subject to appeal before the High Court.

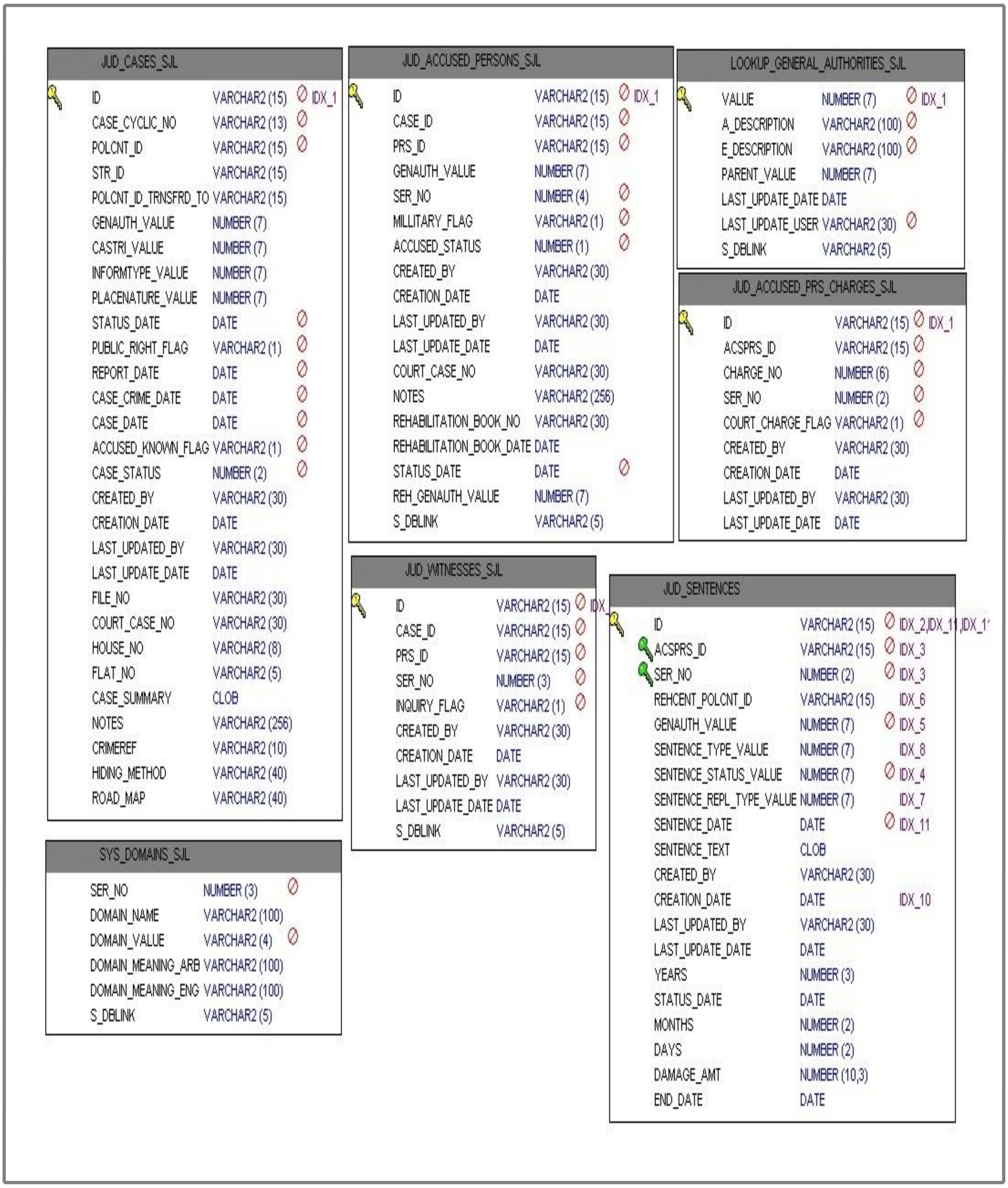
Family protection programmes seek to improve protection of women and children's rights within national judicial and security institutions. These programmes typically involve a range of activities including raising awareness about gender-based, domestic or sexual violence; establishing specialist family protection units, women police stations or increasing the number of female police officers; establishing shelters for women and children who have suffered or been threatened with violence; building the capacity of the judiciary and providing training in family protection issues; strengthening psycho-social and medical support; and supporting legal reform.

Appendix B: Part of Mizan schema And Police Schema

Mizan schema



Police schema



Appendix C: Virtual Schema Scripts

Mizan virtual schema script:

The following scripts are used to build the virtual schema which include Lawsuit, Court, Charge and judgment classes, and executes the mapping process:

```
Sql > Conn mizan/mizan@orcl
//------------------------------------------- Script for creating lawsuit Class in Mizan schema

Sql > CREATE OR REPLACE VIEW LAWSUIT
                (COURT_CODE, LAWSUIT_SERIAL, LAWSUIT_DATE,
                 SOURCE_ID,    LAWSUIT_STATUS_CODE, LAWSUIT_ID,
                 LAWSUIT_CLASS_CODE)
        AS
        SELECT
                COURT_CASE.COURT_CODE,
                to_char(COURT_CASE.CASE_SERIAL),
               COURT_CASE.CASE_DATE,1,COURT_CASE.STATUS_CODE,
                to_char(COURT_CASE.CASE_ID),CASE_CLAIM.CLASS_CODE
        FROM    COURT_CASE ,CASE_CLAIM
        WHERE COURT_CASE.CASE_ID=CASE_CLAIM.CASE_ID
               AND CASE_CLAIM.IS_MAJOR=1


//------------------------------------------- Script for creating Court Class in Mizan schema

Sql > CREATE OR REPLACE VIEW COURT
                (COURT_CODE, COURT_NAME, SOURCE_ID)
        AS
        SELECT court_code, court_name, 1
        FROM COURT1;


//-------------------------------------- Script for creating Charge Class in Mizan schema

Sql > CREATE OR REPLACE VIEW CHARGES
            (CHARGE_NO, LAWSUIT_ID, NATIONAL_NO,
             SOURCE_ID,DISPOSITION_TYPE_CODE)
        AS
        SELECT   p.CLASS_CODE,to_char(p.CASE_ID),e.NATIONAL_NO,1
                 ,p.DISPOSITION_TYPE_CODE
        FROM     CASE_PARTY_CLAIM p , PARTICIPANT  e
        where      p.PARTICIPANT_ID=e.PARTICIPANT_ID;
//-------------------------------------- Script for creating Judgment Class in Mizan schema
```

```
Sql > CREATE OR REPLACE VIEW JUDGMENTS
                 (DECISION_DESC, LAWSUIT_ID, NATIONAL_NO,
                  JUDGMENT_STATMENT , SOURCE_ID)
      AS
      SELECT  jd.PUBLIC_JUDGMENT, to_char( jd.CASE_ID),
               to_char(e.NATIONAL_NO),
              jd.JUDGMENT,1
       FROM   JUDGMENT_DETAILS jd,  PARTICIPANT  e
       Where   jd.PARTICIPANT_ID=e.PARTICIPANT_ID;



//---------------------------------------- Script for creating person Class in Mizan schema

Sql > CREATE OR REPLACE VIEW PERSONS
(KIND_DESC, LAWSUIT_ID, PERSON_TYPE, SOURCE_ID, NATIONAL_NO)
AS
SELECT 'مدني', to_char(d.CASE_ID),f.VALUE,1,e.NATIONAL_NO
FROM CASE_MAIN_PARTY d,PARTICIPANT e, resource_value f
WHERE d.PARTICIPANT_ID=e.PARTICIPANT_ID
           AND d.PARTY_TYPE_CODE=f.KEY
           AND f.ALTERNATIVE_CODE=0 ;
```

Police virtual schema script:

The following scripts are used to build the virtual schemas which include Lawsuit, Court, Charge and judgment classes:

```
Sql > Conn police/police@orcl

//------------------------------------------- Script for creating lawsuit Class in Police schema

Sql > CREATE OR REPLACE VIEW LAWSUIT
        (COURT_CODE, LAWSUIT_SERIAL, LAWSUIT_DATE, SOURCE_ID,
         LAWSUIT_STATUS_CODE, LAWSUIT_ID, LAWSUIT_CLASS_CODE)
         AS
         SELECT  c.GENAUTH_VALUE,c.court_case_no,
                 c.CASE_DATE,2,c.CASE_STATUS,c.ID, null
         FROM    JUD_CASES_SJL c;

//------------------------------------------- Script for creating Court Class in Police schema

Sql > CREATE OR REPLACE VIEW COURT
                 (COURT_CODE, COURT_NAME, SOURCE_ID)
         AS
         SELECT value,A_DESCRIPTION,2
         FROM LOOKUP_GENERAL_AUTHORITIES_SJL;

//------------------------------------------- Script for creating Charge Class in Police schema

Sql > CREATE OR REPLACE VIEW CHARGES
                 (CHARGE_NO, LAWSUIT_ID, NATIONAL_NO, SOURCE_ID,
                 DISPOSITION_TYPE_CODE,  DECISION_DURATION, FINE)
         AS
         SELECT A.CHARGE_NO,J.CASE_ID,O.NATIONAL_ID_NO,2,0,0,0
         FROM JUD_ACCUSED_PRS_CHARGES_SJL A,
               JUD_ACCUSED_PERSONS_SJL J , SYS_PERSONS_SJL O
        WHERE J.ID=A.ACSPRS_ID  AND J.PRS_ID=O.ID;
```

```
//----------------------------------------- Script for creating Judgment Class in Police schema

Sql > CREATE OR REPLACE VIEW JUDGMENTS
       (DECISION_DESC,LAWSUIT_ID ,NATIONAL_NO, judgment_statment ,
        SOURCE_ID  )

     AS
     SELECT   L.A_DESCRIPTION, B.CASE_ID,
              O.NATIONAL_ID_NO , 'غرامة بقيمة '|| D.FINE_AMT||' دنانير'|| D.YEARS
              ||' سنة'|| D.MONTHS ||' شهر '||D.DAYS ||'يوم'  judgment_statment,2
     FROM  JUD_ACCUSED_PERSONS_SJL b,JUD_SENTENCES
              d,LOOKUP_SENTENCE_TYPES_SJL L, SYS_PERSONS_SJL O
     WHERE  B.ID=D.ACSPRS_ID
                AND L.VALUE=D.SENTENCE_TYPE_VALUE
                 AND B.PRS_ID=O.ID;

//----------------------------------------- Script for creating person Class in Police schema

Sql> CREATE OR REPLACE VIEW PERSONS
(KIND_DESC, LAWSUIT_ID, PERSON_TYPE, SOURCE_ID, NATIONAL_NO)
AS
SELECT  'مدني',M.CASE_ID,'مشتكي عليه',2,O.NATIONAL_ID_NO
FROM     JUD_ACCUSED_PERSONS_SJL M , SYS_PERSONS_SJL O
WHERE M.PRS_ID=O.ID
UNION
(
SELECT  'مدني ',M.CASE_ID,'شاهد',2,O.NATIONAL_ID_NO
 FROM     JUD_WITNESSES_SJL M , SYS_PERSONS_SJL O
 WHERE M.PRS_ID=O.ID
UNION
SELECT  'مدني ',M.CASE_ID,'مشتكي',2,O.NATIONAL_ID_NO
 FROM     JUD_VICTIMS_SJL M ,  SYS_PERSONS_SJL O
 WHERE M.PRS_ID=O.ID
);
```

Appendix D: Template Scripts

Mizan Template script:

- The following scripts are used to build the template for Mizan schema:

```
Sql > Conn Mizan/Mizan@orcl

//-------------------script for creating table of schema primary keys

CREATE TABLE TABLES_PKS
AS
SELECT b.table_name,b.constraint_name,c.column_id,b.column_name
FROM user_constraints a, user_cons_columns b ,user_tab_columns c
WHERE a.constraint_type = 'P'
AND a.table_name = b.table_name
AND a.constraint_name = b.constraint_name
AND b.table_name = c.table_name
AND b.column_name = c.column_name
AND b.table_name
IN('CASE_CLAIM','CASE_CLASSIFICATION','CASE_MAIN_PARTY','CASE_PAR
TY_CLAIM','CASE_STATUS','CASE_TYPE','CONSTANTS','COURT','COURT_CA
SE','COURT_CASE_LINK','COURT_DISPOSITION_TYPE','JUDGMENT','JUDGM
ENT_DETAILS','JUDGMENT_TYPE','RESOURCE_VALUE');



//-------------------script for creating table of schema foreign keys

CREATE TABLE TABLES_FKS
AS
SELECT b.table_name,b.constraint_name,c.column_id,b.column_name,
            a. r_constraint_name
FROM user_constraints a, user_cons_columns b ,user_tab_columns c
WHERE a.constraint_type = 'R'
AND a.table_name = b.table_name
AND a.constraint_name = b.constraint_name
AND b.table_name = c.table_name
AND b.column_name = c.column_name
AND b.table_name
IN('CASE_CLAIM','CASE_CLASSIFICATION','CASE_MAIN_PARTY','CASE_PAR
TY_CLAIM','CASE_STATUS','CASE_TYPE','CONSTANTS','COURT','COURT_CA
SE','COURT_CASE_LINK','COURT_DISPOSITION_TYPE','JUDGMENT','JUDGM
ENT_DETAILS','JUDGMENT_TYPE','RESOURCE_VALUE');
```

```
//-------------------script for creating table of schema tables

CREATE TABLE TABLES_names
AS
SELECT Distinct   table_name
FROM TABS
//-------------------script for creating table of schema tables columns

CREATE TABLE TABLES_COULMNS
AS
SELECT a.table_name ,a.column_name , a.data_type ,a.data_length , a.data_precision,
            a.data_scale
FROM user_tab_columns a
where a.table_name
IN('CASE_CLAIM','CASE_CLASSIFICATION','CASE_MAIN_PARTY','CASE_PAR
TY_CLAIM','CASE_STATUS','CASE_TYPE','CONSTANTS','COURT','COURT_CA
SE','COURT_CASE_LINK','COURT_DISPOSITION_TYPE','JUDGMENT',
'JUDGMENT_DETAILS','JUDGMENT_TYPE','RESOURCE_VALUE','')
ORDER BY a.column_id ASC;
```

Police Template script:

- The following scripts are used to build the template for police schema:

Sql > Conn police/police@orcl

```
//-------------------script for creating table of schema primary keys

CREATE TABLE TABLES_PKS
AS
SELECT b.table_name,b.constraint_name,c.column_id,b.column_name
FROM user_constraints a, user_cons_columns b ,user_tab_columns c
WHERE a.constraint_type = 'P'
AND a.table_name = b.table_name
AND a.constraint_name = b.constraint_name
AND b.table_name = c.table_name
AND b.column_name = c.column_name
AND b.table_name IN('JUD_CASES_SJL','JUD_ACCUSED_PERSONS_SJL',
'JUD_SENTENCES' ,'SYS_PERSONS_SJL' , 'LOOKUP_SENTENCE_TYPES_SJL',
'JUD_ACCUSED_PRS_CHARGES_SJL','LOOKUP_GENERAL_AUTHORITIES_SJ
L');

//-------------------script for creating table of schema foreign keys

CREATE TABLE TABLES_FKS
AS
SELECT b.table_name,b.constraint_name,c.column_id,b.column_name,
              a. r_constraint_name
FROM user_constraints a, user_cons_columns b ,user_tab_columns c
WHERE a.constraint_type = 'R'
AND a.table_name = b.table_name
AND a.constraint_name = b.constraint_name
AND b.table_name = c.table_name
AND b.column_name = c.column_name
AND b.table_name IN('JUD_CASES_SJL','JUD_ACCUSED_PERSONS_SJL',
'JUD_SENTENCES' ,'SYS_PERSONS_SJL' , 'LOOKUP_SENTENCE_TYPES_SJL',
'JUD_ACCUSED_PRS_CHARGES_SJL','LOOKUP_GENERAL_AUTHORITIES_SJ
L');
```

```
//-------------------script for creating table of schema tables

CREATE TABLE TABLES_names
AS
SELECT Distinct   table_name
FROM TABS


//-------------------script for creating table of schema tables columns

CREATE TABLE TABLES_COULMNS
AS
SELECT a.table_name ,a.column_name , a.data_type ,a.data_length , a.data_precision,
            a.data_scale
FROM user_tab_columns a
where a.table_name IN('JUD_CASES_SJL','JUD_ACCUSED_PERSONS_SJL',
'JUD_SENTENCES' ,'SYS_PERSONS_SJL' , 'LOOKUP_SENTENCE_TYPES_SJL',
'JUD_ACCUSED_PRS_CHARGES_SJL','LOOKUP_GENERAL_AUTHORITIES_SJ
L');
ORDER BY a.column_id ASC;
```

Appendix E: Schema DDL Auditing Script

- The following script are used to build auditing table that register each change may be occurred on database schema:

```
//------------------- DDl trigger script for DDL auditing:

//-------------------step 1: create auditing table

Sql> CREATE TABLE DDLAuditing (
 DateOfChange DATE,
 OSUSER  VARCHAR2(255),
 CURRENT_USER   VARCHAR2(255),
 HOST   VARCHAR2(255),
 TERMINAL   VARCHAR2(255),
  owner   VARCHAR2(30),
  TYPE   VARCHAR2(30),
  name   VARCHAR2(30),
  sysevent VARCHAR2(30));

//-------------------step 2: create auditing triggers

Sql> CREATE OR REPLACE TRIGGER audit_ddl_trg AFTER ddl ON SCHEMA
BEGIN
        IF (ora_sysevent='TRUNCATE')
        THEN
         NULL; -- I do not care about truncate
       ELSE
       INSERT INTO DDLAuditing
         (DateOfChange,osuser,current_user,host,terminal,owner,TYPE,name,sysevent)
       VALUES(
        SYSDATE,
         sys_context('USERENV','OS_USER') ,
         sys_context('USERENV','CURRENT_USER') ,
         sys_context('USERENV','HOST') ,
         sys_context('USERENV','TERMINAL') ,
         ora_dict_obj_owner,
         ora_dict_obj_type,
         ora_dict_obj_name,
         ora_sysevent    );
 END IF;
END;
```

Appendix F: Unification Schema Script

```
//----------------------Script for creating LAWSUIT Class in unification virtual schema
Sql > CREATE OR REPLACE VIEW LAWSUIT
      (COURT_CODE, LAWSUIT_SERIAL, LAWSUIT_DATE, SOURCE_ID,
       LAWSUIT_STATUS_CODE, LAWSUIT_ID, LAWSUIT_CLASS_CODE)
       AS
       SELECT  * from  POLICE. LAWSUIT
        UNION
       (SELECT  * from MIZAN. LAWSUIT)

//------------------------ Script for creating COURT Class in unification virtual schema

Sql > CREATE OR REPLACE VIEW COURT
       (COURT_CODE, COURT_NAME, SOURCE_ID)
       AS
       SELECT * FROM   POLICE.COURT
       UNION
       ( SELECT * FROM   MIZAN.COURT );

//--------------------- Script for creating CHARGES Class in unification virtual schema
Sql >CREATE OR REPLACE VIEW CHARGES
           (CHARGE_NO, LAWSUIT_ID, NATIONAL_NO,
            SOURCE_ID,DISPOSITION_TYPE_CODE)
       AS
       SELECT * FROM   POLICE. CHARGES
       UNION
       ( SELECT * FROM   MIZAN. CHARGES );

//------------------------ Script for creating Judgments Class in unification virtual schema

Sql > CREATE OR REPLACE VIEW JUDGMENTS
(DECISION_DESC,LAWSUIT_ID   ,NATIONAL_NO,
judgment_statment     ,SOURCE_ID )
 AS
SELECT *  FROM   POLICE. JUDGMENTS
UNION
(
SELECT *  FROM   MIZAN. JUDGMENTS
);
//------------------------ Script for creating  person Class in unification virtual schema

Sql> CREATE OR REPLACE VIEW PERSONS
(KIND_DESC, LAWSUIT_ID, SOURCE_ID, NATIONAL_NO)
AS
SELECT   * FROM   POLICE.PERSONS
UNION
(
SELECT  *  FROM   MIZAN.PERSONS );
```

إيجاد قواعد بيانات متكاملة في الشبكات التعاونية

إعداد
نانسي جميل خريس

المشرف
الأستاذ الدكتور نديم عبيد

ملخص

إن معظم أنظمة المعلومات الحاسوبية في المؤسسات الكبيرة اليوم يتم بناؤها بشكل موزع في أماكن مختلفة من المؤسسة حيث يتم تصميم كل نظام معلوماتي وتطويره وإدارته بشكل مستقل. هذه العوامل أدت إلى ظهور أنواع مختلفة من عدم التجانس: كعدم التجانس النحوي والدلالي والهيكلي مما جعل عملية تجميع البيانات وإظهارها بشكل موحد أكثر صعوبة. في ضوء هذا الوضع ،اقترحنا نهج جديد قمنا بتجميعه من مناهج أخرى مستخدمة في عملية تجميع البيانات يهدف إلى إمكانية تفاعل قواعد البيانات مع بعضها البعض وحل مشاكل عدم التجانس بشكل أكثر فاعلية باستخدام فكرة الانتولوجيا (Ontology) وقالب المخطط (Template Schema).

يستند النهج الجديد المقترح على أربع عمليات رئيسية: رسم الخرائط والتوحيد وتدقيق أوامر لغة تعريف البيانات (DDL) وتبادل المخططات باستخدام قالب المخطط.

رسم الخرائط هي عملية شبه آلية تستخدم الانتولوجيا لربط مخطط مصدر قاعدة البيانات مع الإطار المفاهيمي المدعى المخطط الوهمي. التوحيد هي عملية مؤتمتة لدمج المخططات الوهمية بالتالي قواعد البيانات التي ترتبط بها. أما تدقيق أوامر الـــ DDL فهي عملية آلية لتسجيل التغييرات التي قد تحدث في بنية مصدر البيانات وإبلاغ مسؤول النظام بوجود بعض المشاكل التي يجب معالجتها. يستخدم قالب المخطط لتمثيل فئة من مخططات قواعد البيانات المختلفة. إن الهدف من استخدام قالب المخطط هو إيجاد آلية لترجمة أي مخطط قاعدة بيانات متوافق مع قالب المصدر في شكل مطابق للقالب الهدف. لقد قمنا بتطبيق هذا النهج الجديد لتجميع مصدران مختلفان للبيانات يتعلقان بالمجال القانوني وهما قاعدة بيانات ميزان – النظام المستخدم في وزارة العدل– وقاعدة بيانات الشرطة لمعرفة السيرة الجرمية لكل مواطن في هذا البلد. بالإضافة إلى ذلك، قمنا بدمج نظام الأحوال المدنية مع نظامنا للحصول على أحدث البيانات الشخصية عن كل مواطن .